HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

www.heig-vd.ch

03 - Business tier

EJBs, managed objects, AOP, dependency injection, object pooling

AMT 2019

Olivier Liechti

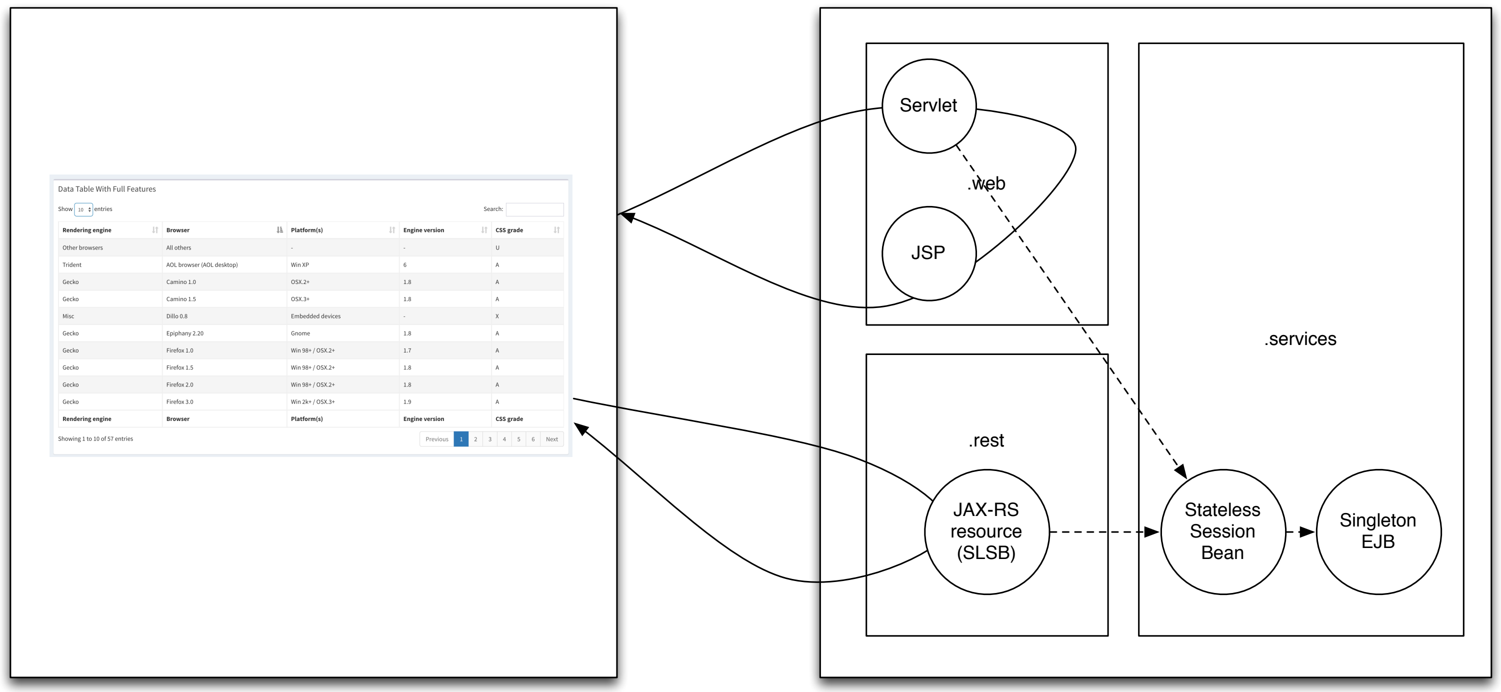
heig-vd HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD
www.heig-vd.ch

The business tier: EJBs as an example of “managed components”
Break
Introduction to load testing with JMeter
Java EE mechanisms from the ground up: the AMT “application server”

Webcasts

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

1 MVC - the browser asks for a an HTML page and its assets (css, js, etc.)



Data Table With Full Features

Show 10 entries Search:

Rendering engine	Browser	Platform(s)	Engine version	CSS grade
Other browsers	All others	-	-	U
Trident	AOL browser (AOL desktop)	Win XP	6	A
Gecko	Camino 1.0	OSX.2+	1.8	A
Gecko	Camino 1.5	OSX.3+	1.8	A
Misc	Dillo 0.8	Embedded devices	-	X
Gecko	Epiphany 2.20	Gnome	1.8	A
Gecko	Firefox 1.0	Win 98+ / OSX.2+	1.7	A
Gecko	Firefox 1.5	Win 98+ / OSX.2+	1.8	A
Gecko	Firefox 2.0	Win 98+ / OSX.2+	1.8	A
Gecko	Firefox 3.0	Win 2k+ / OSX.3+	1.9	A
Rendering engine	**Browser**	**Platform(s)**	**Engine version**	**CSS grade**

Showing 1 to 10 of 57 entries

Previous 1 2 3 4 5 6 Next

2 SPA - the data tables script invokes the REST API to get data (AJAX)

Tasks

1. Create a new project
- 1.1. the code deployed in Glassfish and Wildfly will be slightly different
- 1.2. for this reason, we will work in 2 branches: fb-rest-glassfish, fb-rest-wil

2. Implement the business services layer with EJBs
- 2.1. Implement a singleton EJB
- 2.2. Implement a stateless session bean
- 2.3. Inject the stateless session bean in a servlet

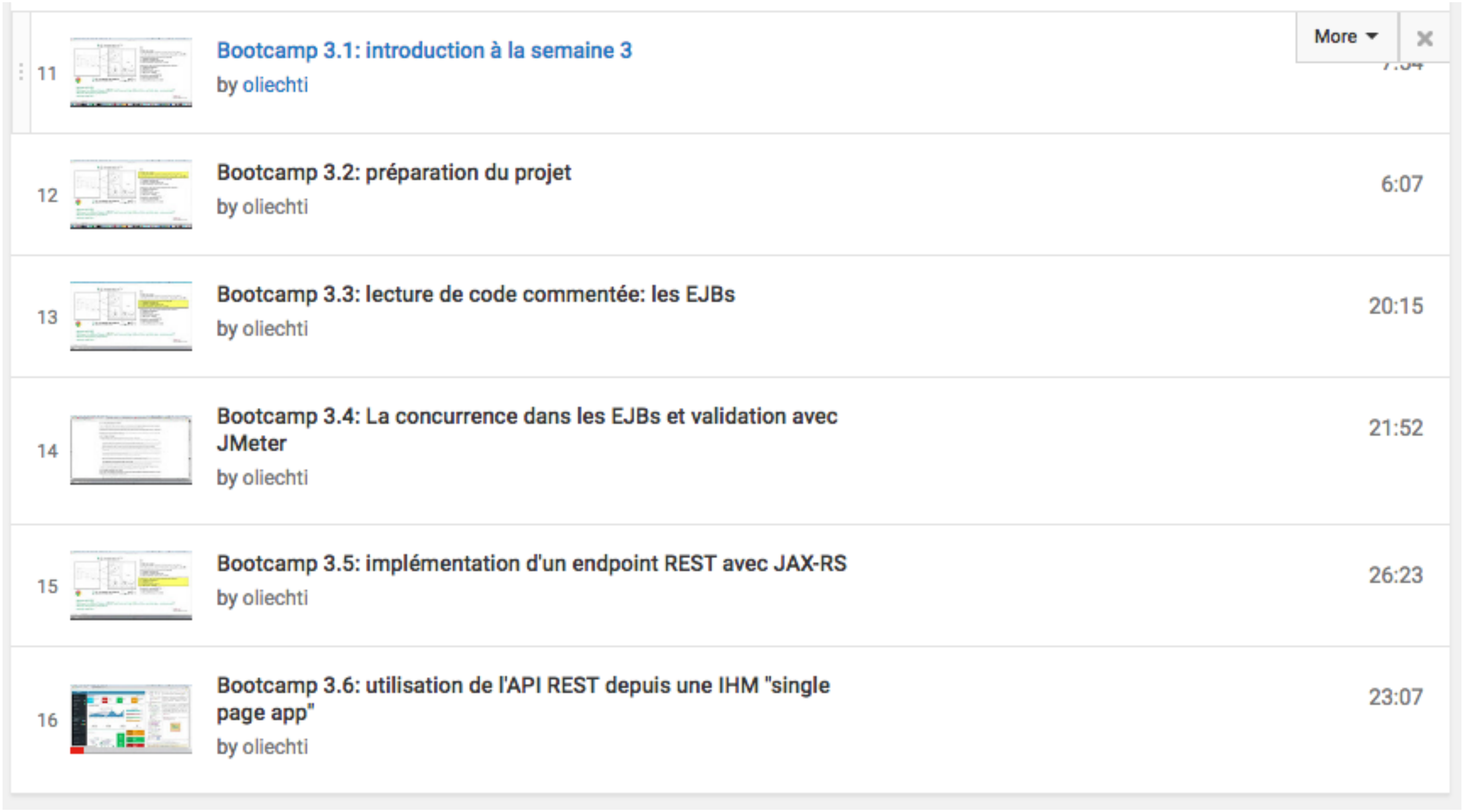
3. Implement a REST API with JAX-RS (Jersey and Jackson)
- 3.1. Configure the framework
- 3.2. Implement DTOs
- 3.3. Implement a REST endpoint
- 3.4. Test the REST endpoint

4. Build a UI on top of the REST API
- 4.1. Select and study a template
- 4.2. Discover jquery datatables
- 4.3. Integrate the template in the project

What is an EJB?
What are the different types of EJBS and how are they different from servlets (e.g. concurrency)?
What is dependency injection?
What is JAX-RS?

Webcasts

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



More

11 Bootcamp 3.1: introduction à la semaine 3
by oliechti

12 Bootcamp 3.2: préparation du projet
by oliechti
6:07

13 Bootcamp 3.3: lecture de code commentée: les EJBs
by oliechti
20:15

14 Bootcamp 3.4: La concurrence dans les EJBs et validation avec JMeter
by oliechti
21:52

15 Bootcamp 3.5: implémentation d'un endpoint REST avec JAX-RS
by oliechti
26:23

16 Bootcamp 3.6: utilisation de l'API REST depuis une IHM "single page app"
by oliechti
23:07

Warning!

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD
www.heig-vd.ch

The webcast was recorded for another edition of the course.
This year, the planning is a bit different.

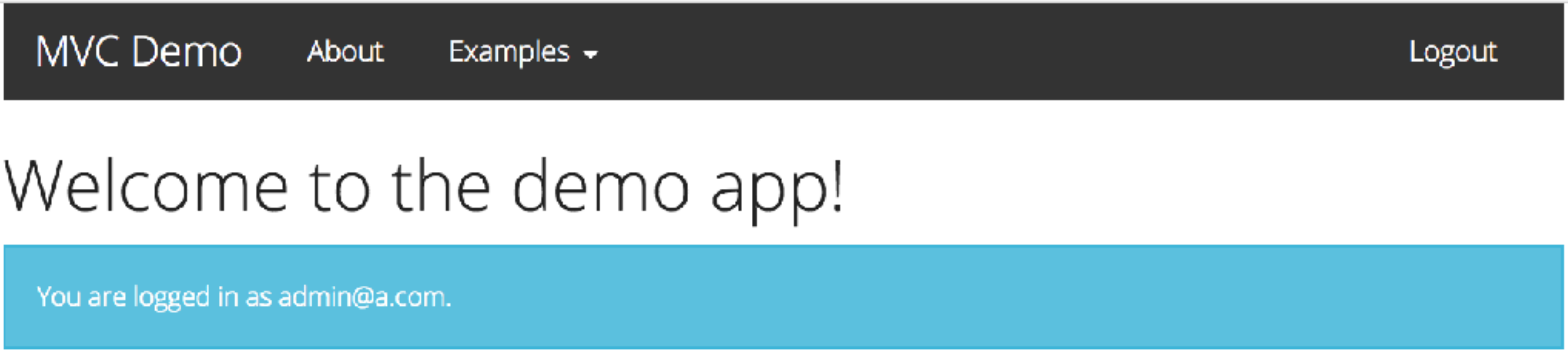
Some of the topics will be presented later:

- REST APIs with JAX-RS
- Data Transfer Objects (DTOs)
- Single Page App

MVC demo application

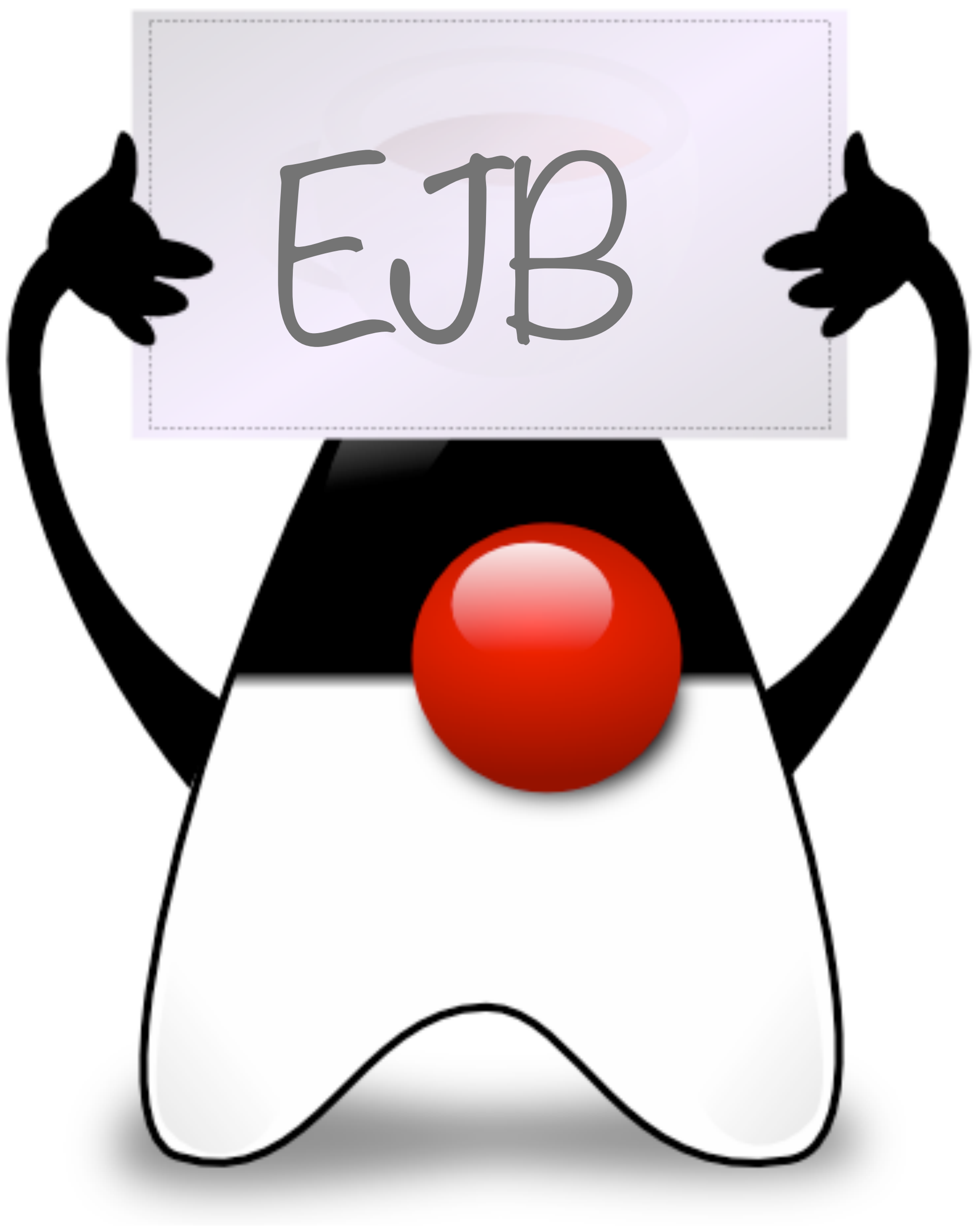
https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-AMT-Example-MVC

checkout MVC-EJB-Concurrency-NoDB



Welcome to the demo app!

You are logged in as admin@a.com.



HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

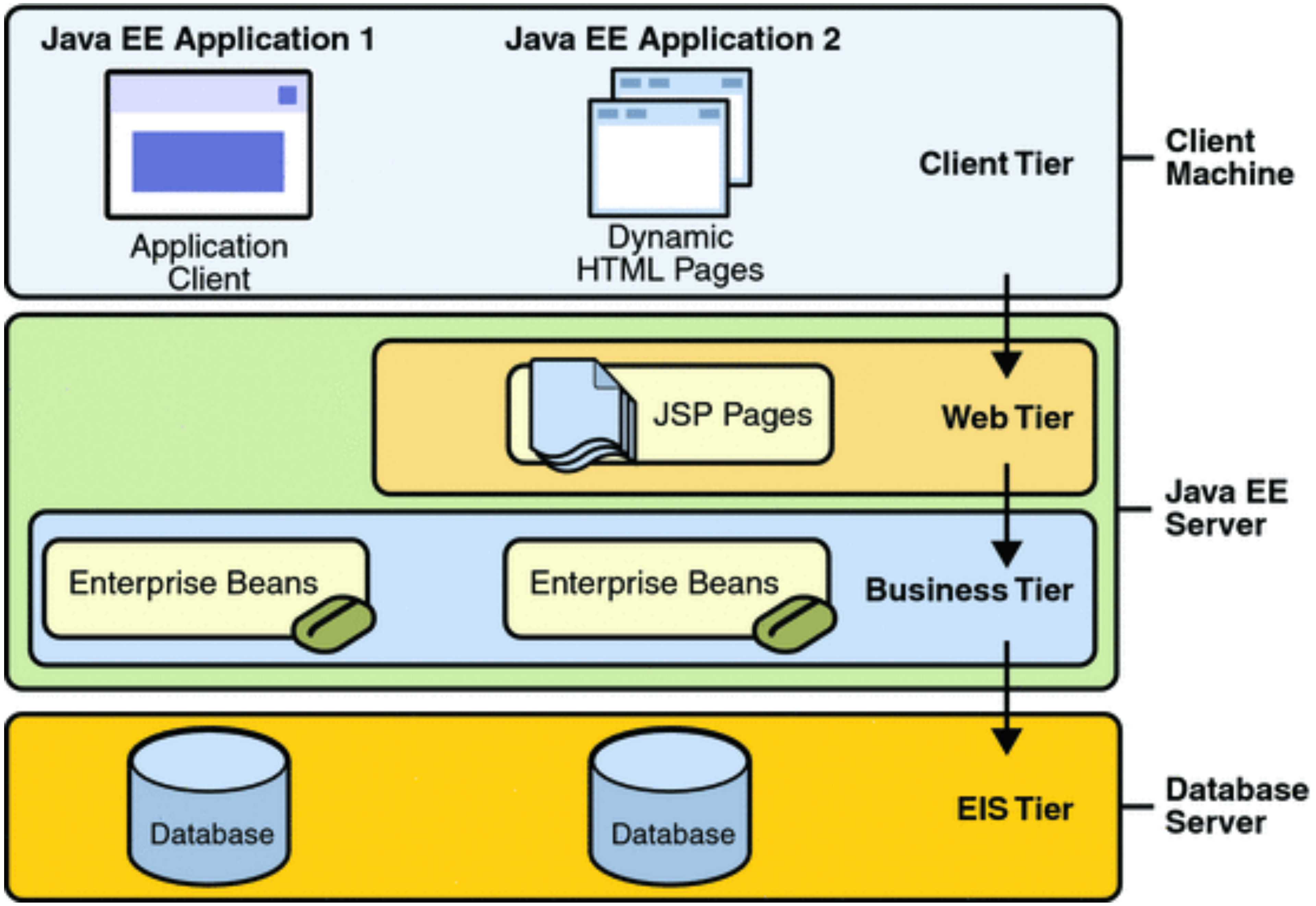
Business Services & EJB

Services in a Java EE application

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- Last week, we implemented a very simple Java EE application.
- When we implemented the MVC pattern, we implemented a service as a **Plain Old Java Object** (POJO).
- **The POJO was not a managed component.** We created the instance(s) of the service (*in the web container*).
- This week, we will see an **alternative solution** for implementing Java EE services: Enterprise Java Beans (EJBs).

What is the best way to implement services, POJOs or EJBs? **There is not a single right answer to this question!** There are pros and cons in both approach.



http://java.sun.com/javaee/5/docs/tutorial/doc/bnaay.html

What is an **Enterprise Java Bean** (EJB)?

- An EJB is a **managed component**, which implements **business logic** *in a UI agnostic way*.
- The EJB container manages the **lifecycle** of the EJB instances.
- The EJB container also **mediates the access** from clients (i.e. it is an “invisible” intermediary) to EJBs. This is a form of Aspect Oriented Programming (AOP):
- This allows the EJB container to perform technical operations (especially related to **transactions** and **security**) when EJBs are invoked by clients.
- The EJB container manages a **pool** of EJB instances.
- Note: the EJB 3.2 API is **specified is JSR 345**.

What are the **4 types** of EJBs used today?

- **Stateless Session Beans** are used to implement business services, where every client request is independent.
- **Stateful Session Beans** are used for services which have a notion of conversation (e.g. shopping cart).
- **Singleton Session Beans** are used when there should be a single service instance in the app.
- **Message Driven Beans** are used together with the Java Message Service (JMS). Business logic is not invoked when a web client sends a request, but when a message arrives in a queue. We will see that later.

When you implement a stateful application in Java EE, **you have the choice to store the state in different places**. One option is to do it in the web tier (in the HTTP session). Another option is to use **Stateful Session Beans**. Many (most) developers use HTTP sessions.

In older versions of Java EE (before Java EE 5), there was another type of EJBs: **Entity Beans**.

Entity Beans were used for **accessing the database**. They were a nightmare to use and raised a number of issues. You might find them in legacy applications.

Entity Beans (as a legacy type of EJB) are **not the same thing** as **JPA Entities**, which are now widely used!

A first example

```
package ch.heigvd.amt.lab1.services;
import javax.ejb.Local;

@Local
public interface CollectorServiceLocal {

  void submitMeasure(Measure measure);

}
```

```
package ch.heigvd.amt.lab1.services;
import javax.ejb.Stateless;

@Stateless
public class CollectorService implements CollectorServiceLocal {

  @Override
  public void submitMeasure(Measure measure) {
    // do something with the measure (process, archive, etc.)
  }

}
```

These **annotations** are processed by the application server at **deployment time**.

They are an **declaration** that the service must be handled as a **managed component**!

How does a “client” find and use an EJB?

- By “**client**”, we refer to a **Java component** that wants to get a reference to the EJB and invoke its methods.
- In many cases, the client is a **servlet** or **another EJB** (i.e. a service that delegates part of the work to another service).
- The application server is providing a **naming and directory service** for managed components. Think of it as a “white pages” service that keeps track of component names and references.
- Remember that we mentioned **Dependency Injection** earlier today?

The Java Naming and Directory Interface (JNDI) provides an API to access directory services. It can be used to access an LDAP server. It can also be used to lookup components in a Java EE server.

The **first method** to find an EJB is to do an **explicit lookup**, with JNDI.

```
@WebServlet(name = "FrontController", urlPatterns = {"/FrontController"})
public class FrontController extends HttpServlet {

  private CollectorServiceLocal collectorService;

  @Override
  public void init() throws ServletException {
    super.init();
    try {
     Context ctx = new InitialContext();
     collectorService = (CollectorServiceLocal) ctx.lookup("java:module/CollectorService");
    } catch (NamingException ex) {
      Logger.getLogger(FrontController.class.getName()).log(Level.SEVERE, null, ex);
    }
  }
```

This gives me access to the app server's naming service

I am using the app server's naming service

Warning! These 2 JNDI operations are **costly** (performance-wise). You don't want to re-execute them for every single HTTP request!!!!
It is much better to do it once and to **cache the references** to the services.

The **second method** is to ask the app server to **inject a dependency** to the service.

```
@WebServlet(name = "FrontController", urlPatterns = {"/FrontController"})
public class FrontController extends HttpServlet {

  @EJB
  private CollectorServiceLocal collectorService;

}
```

With the @EJB annotation, **I am declaring a dependency** from between my servlet and my service. The servlet *uses* the service.

With the @EJB annotation, I am also giving instructions to the app server. The servlet and the service are **managed components**. When the app server instantiates the servlet, it **injects a value** into the **collectorService** variable.

EJBs in the MVCDemo project

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

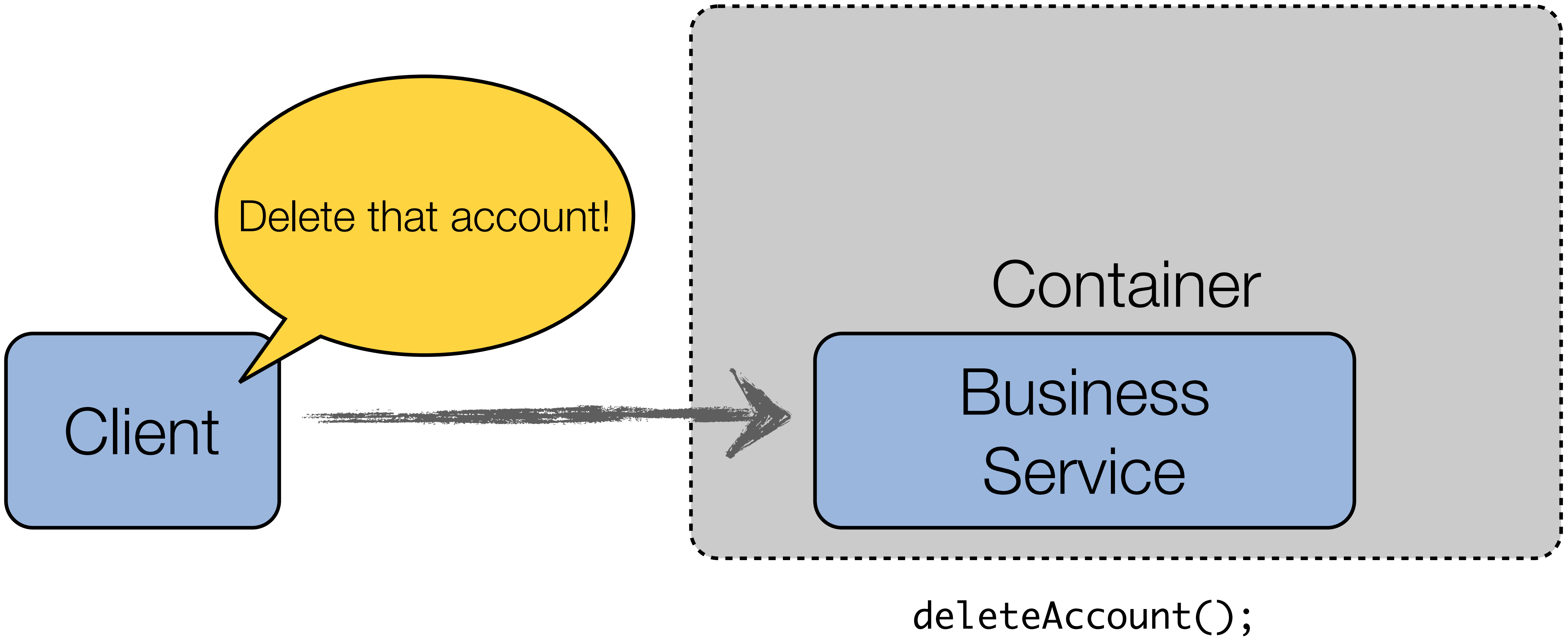
```
@Singleton
public class BeersDataStore implements BeersDataStoreLocal {

 private final List<Beer> catalog = new LinkedList<>();

 public BeersDataStore() {
    catalog.add(new Beer("Cardinal", " Feldschlösschen", "Switzlerland", "Lager"));
    catalog.add(new Beer("Punk IPA", " BrewDog", "Scotland", "India Pale Ale"));
 }
...
}
```

```
@Stateless
public class BeersManager implements BeersManagerLocal {

  @EJB
  BeersDataStoreLocal beersDataStore;

  @Override
  public List<Beer> getAllBeers() {
    simulateDatabaseDelay();
    return beersDataStore.getAllBeers();
  }
...
}
```

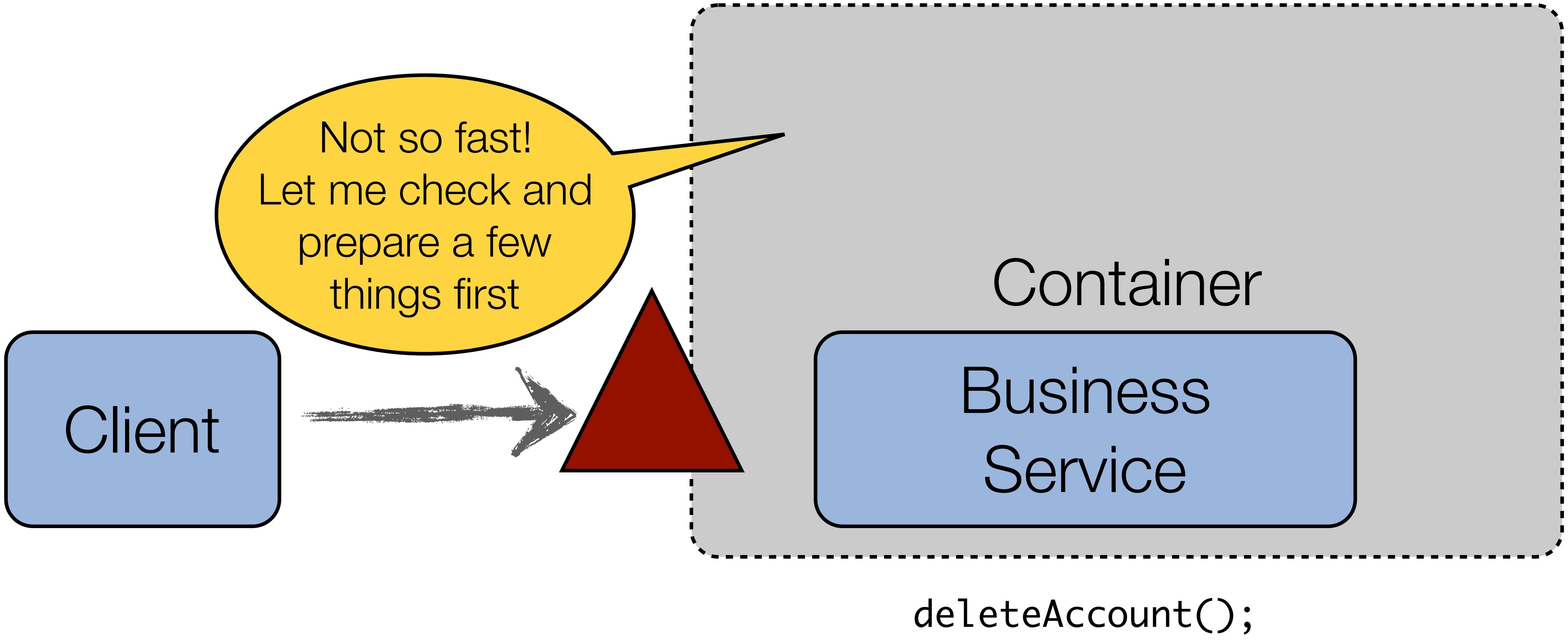
EJBs in the MVCDemo project

```
public class BeersServlet extends HttpServlet {

  @EJB
  BeersManagerLocal beersManager;

  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
     throws ServletException, IOException {
    /*
     Firstly, we need to get a model. It is not the responsibility of the servlet
     to build the model. In other words, you should avoid to put business logic
     and database access code directly in the controller. In this example, the
     beersManager takes care of the model construction.
     */
    Object model = beersManager.getAllBeers();
    ...
   }
   ...
}
```

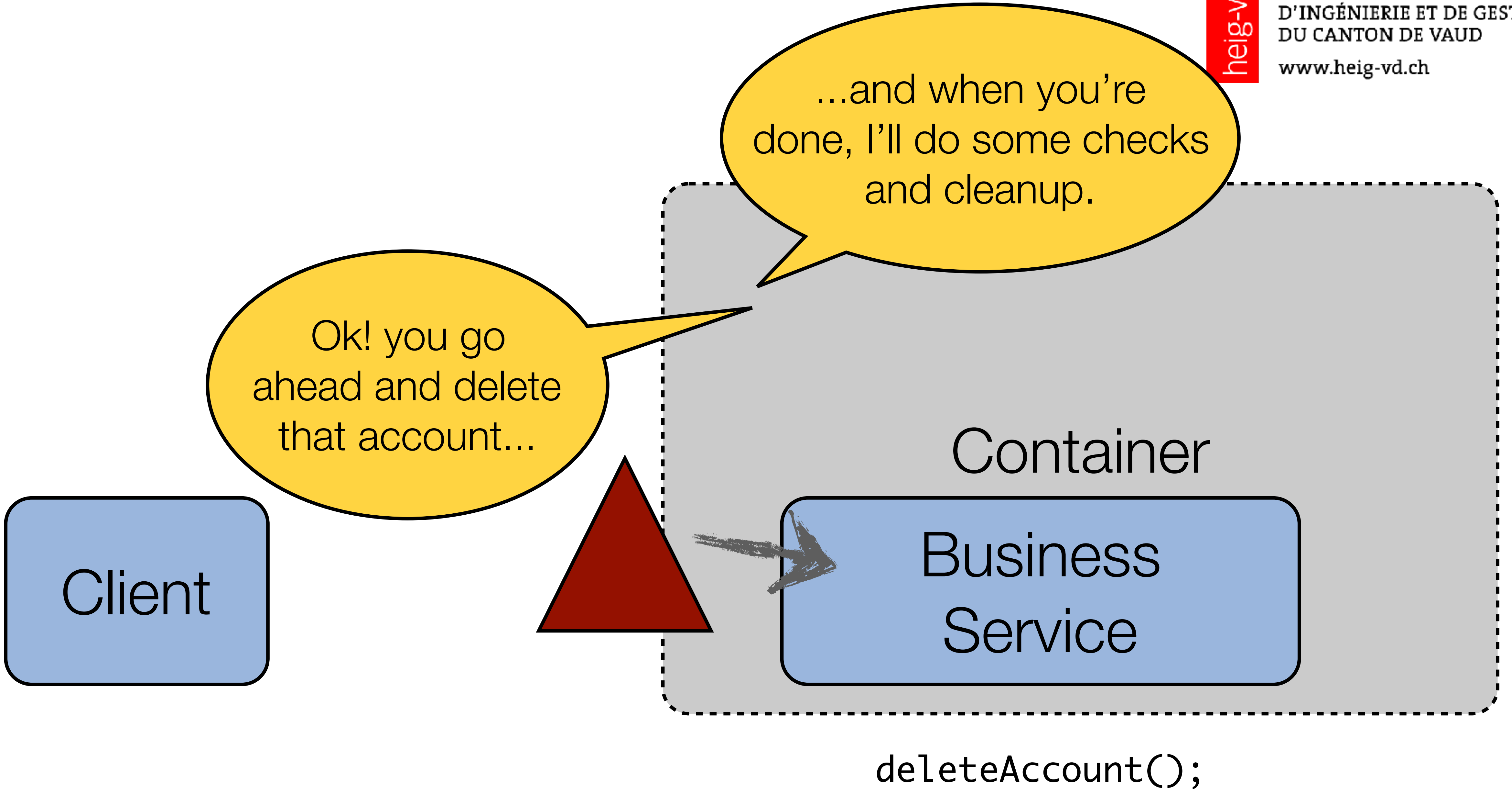
The app server **mediates** the access between clients and EJBs. What does it mean?



The business service, implemented as a Stateless Session Bean, is a **managed component**.
The client ***thinks*** that he has a direct reference to a Java object.
He is ***wrong***.



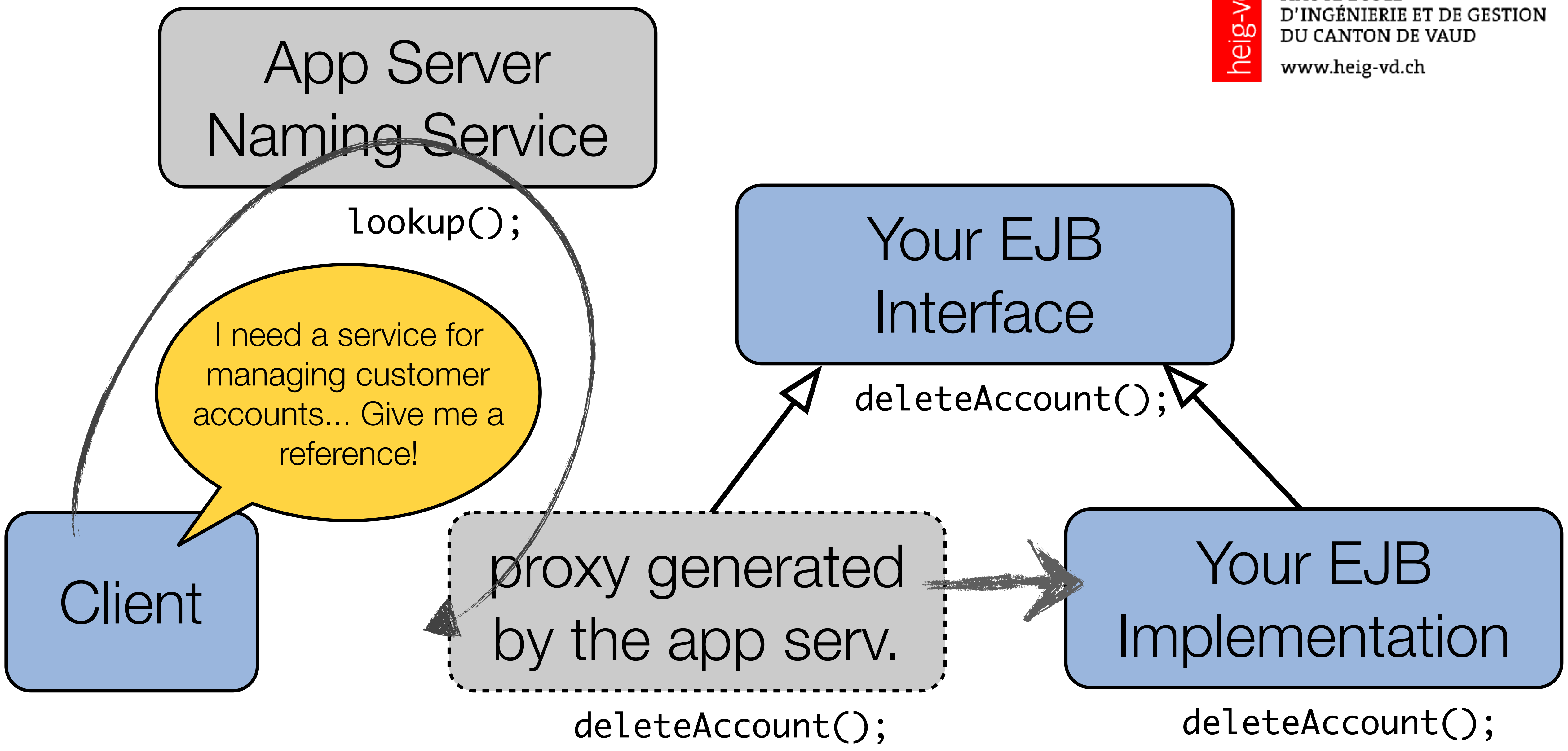
In reality, when the client invokes the deleteAccount() methods, the call is going **through the container**.
The container is in a position to **perform various tasks** (security checks, transaction demarcation, etc.)


...and when you're done, I'll do some checks and cleanup.
Ok! you go ahead and delete that account...
Container
Client
Business Service
deleteAccount();

When done, the container can forward the method call to the business service (your implementation).
On the way back, the response also goes back **via the container**.

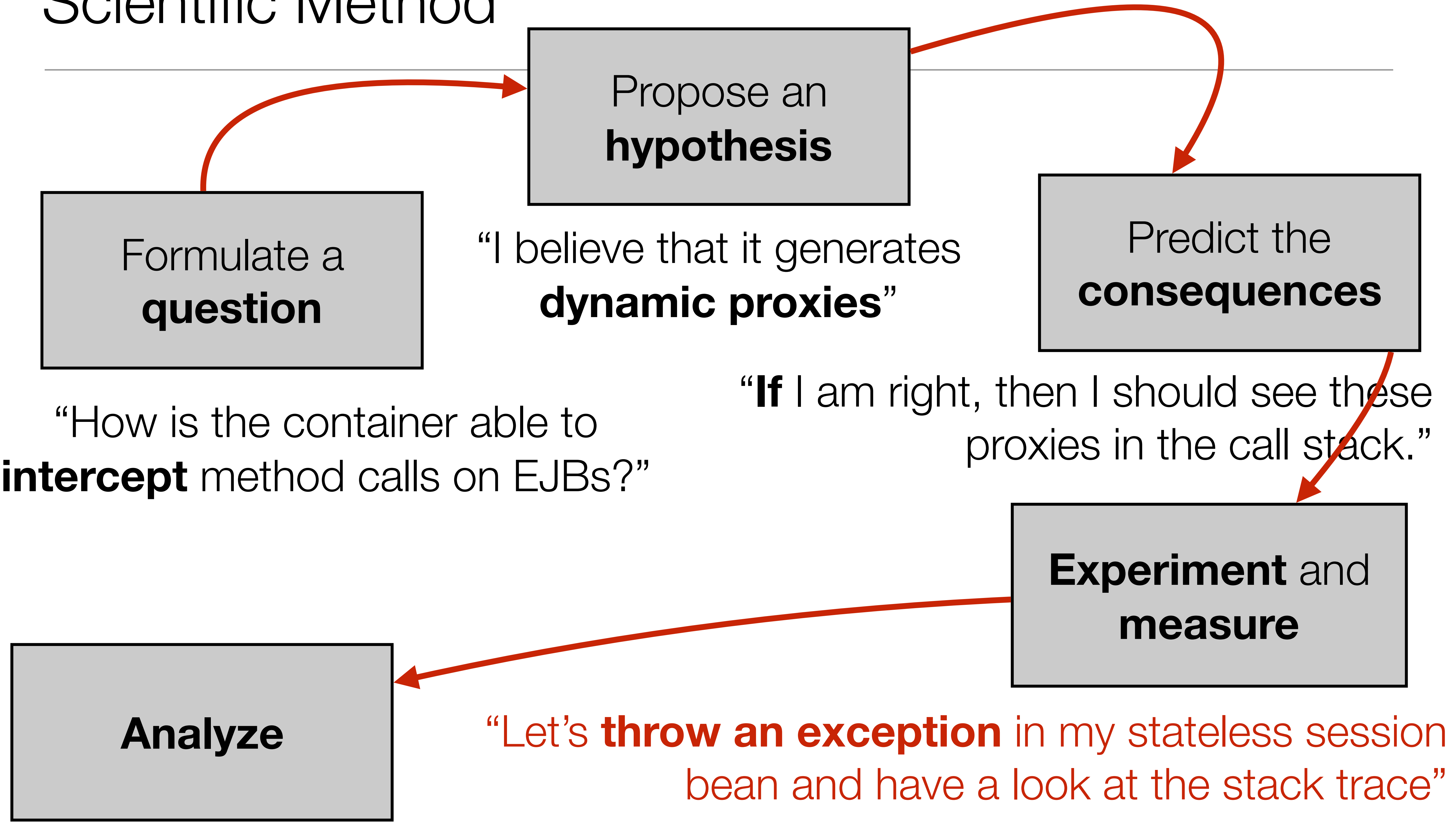
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

How is that possible?
How does it work?


App Server Naming Service
lookup();
I need a service for managing customer accounts... Give me a reference!
Client
proxy generated by the app serv.
deleteAccount();
Your EJB Interface
deleteAccount();
Your EJB Implementation
deleteAccount();

Your service implementation implements your interface.
The container dynamically generates a class, which implements the same interface. This class performs the technical tasks and invokes your class (proxy).

Scientific Method


Formulate a **question**
Propose an **hypothesis**
Predict the **consequences**
Experiment and **measure**
Analyze

"How is the container able to **intercept** method calls on EJBs?"

"I believe that it generates **dynamic proxies**"

"**If** I am right, then I should see these proxies in the call stack."

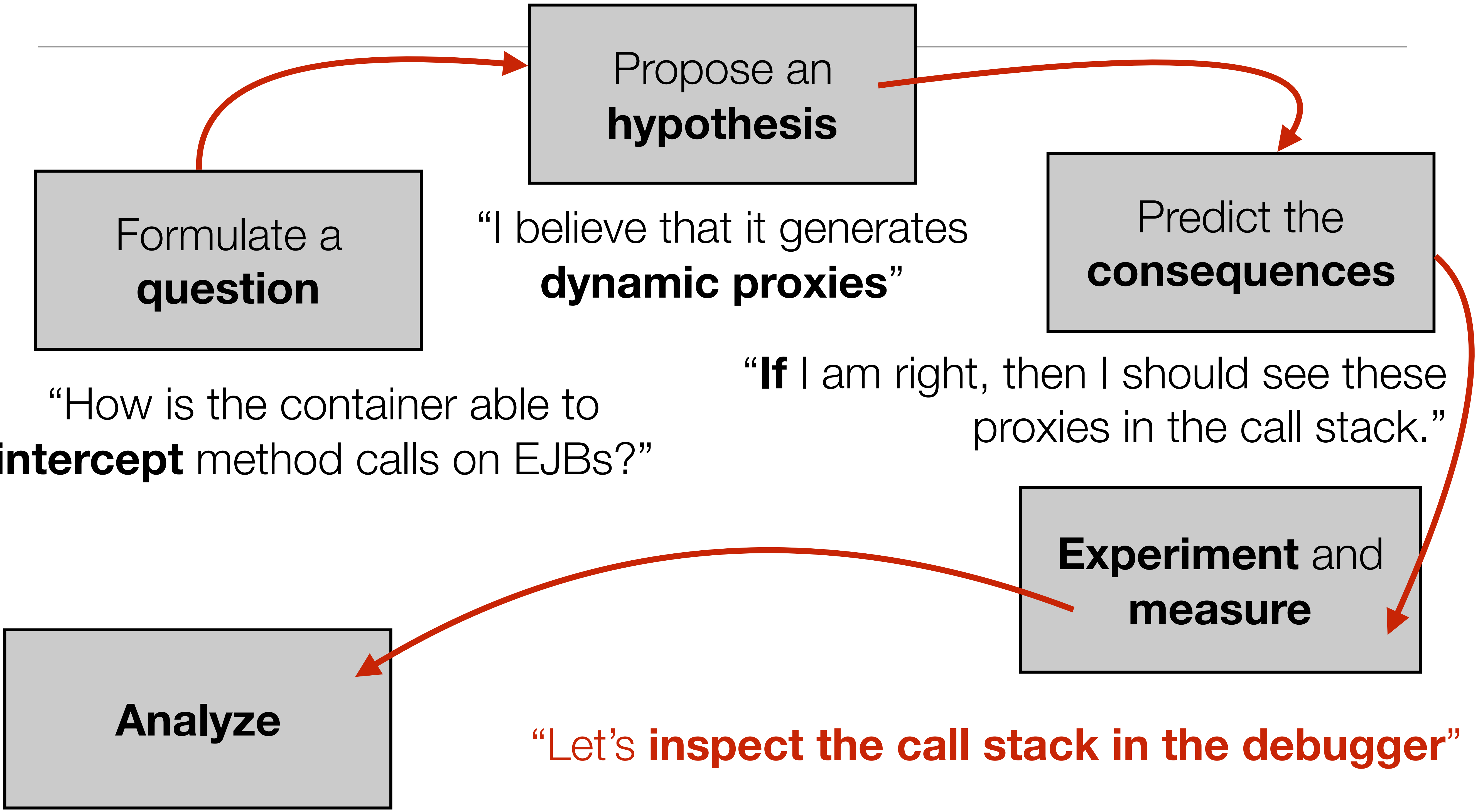
"Let's **throw an exception** in my stateless session bean and have a look at the stack trace"

"In the stack trace, I can confirm that the servlet is not directly calling my Stateless Session Bean implementation class."

```
Caused by: java.lang.RuntimeException: just kidding
	at ch.heigvd.amt.lab1.services.CollectorService.submitMeasure(CollectorService.java:15)
	at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
	at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
	at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
	at java.lang.reflect.Method.invoke(Method.java:483)
	at org.glassfish.ejb.security.application.EJBSecurityManager.runMethod(EJBSecurityManager.java:1081)
	at org.glassfish.ejb.security.application.EJBSecurityManager.invoke(EJBSecurityManager.java:1153)
	at com.sun.ejb.containers.BaseContainer.invokeBeanMethod(BaseContainer.java:4786)
	at com.sun.ejb.EjbInvocation.invokeBeanMethod(EjbInvocation.java:656)
	at com.sun.ejb.containers.interceptors.AroundInvokeChainImpl.invokeNext(InterceptorManager.java:822)
	at com.sun.ejb.EjbInvocation.proceed(EjbInvocation.java:608)
	at 
org.jboss.weld.ejb.AbstractEJBRequestScopeActivationInterceptor.aroundInvoke(AbstractEJBRequestScopeActivationInte
ceptor.java:46)
	at org.jboss.weld.ejb.SessionBeanInterceptor.aroundInvoke(SessionBeanInterceptor.java:52)
	at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
	at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethod
	at sun.reflect.DelegatingMethodAccessorImpl.invoke(Delegati
	at java.lang.reflect.Method.invoke(Method.java:483)
	at com.sun.ejb.containers.interceptors.AroundInvokeIntercep
	at com.sun.ejb.containers.interceptors.AroundInvokeChainImp
	at com.sun.ejb.EjbInvocation.proceed(EjbInvocation.java:608
	at com.sun.ejb.containers.interceptors.SystemInterceptorPro
	at com.sun.ejb.containers.interceptors.SystemInterceptorPro
	at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Meth
	at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethod
	at sun.reflect.DelegatingMethodAccessorImpl.invoke(Delegati
	at java.lang.reflect.Method.invoke(Method.java:483)
	at com.sun.ejb.containers.interceptors.AroundInvokeInterceptor.intercept(InterceptorManager.java:883)
	at com.sun.ejb.containers.interceptors.AroundInvokeChainImpl.invokeNext(InterceptorManager.java:822)
	at com.sun.ejb.containers.interceptors.InterceptorManager.intercept(InterceptorManager.java:369)
	at com.sun.ejb.containers.BaseContainer.__intercept(BaseContainer.java:4758)
	at com.sun.ejb.containers.BaseContainer.intercept(BaseContainer.java:4746)
	at com.sun.ejb.containers.EJBLocalObjectInvocationHandler.invoke(EJBLocalObjectInvocationHandler.java:212)
	... 34 more
```

```
@Stateless
public class CollectorService implements Coll

  @Override
  public void submitMeasure(Measure measure)
    throw new RuntimeException("just kidding"
  }

}
```

Scientific Method


Formulate a **question**
Propose an **hypothesis**
Predict the **consequences**
Experiment and **measure**
Analyze

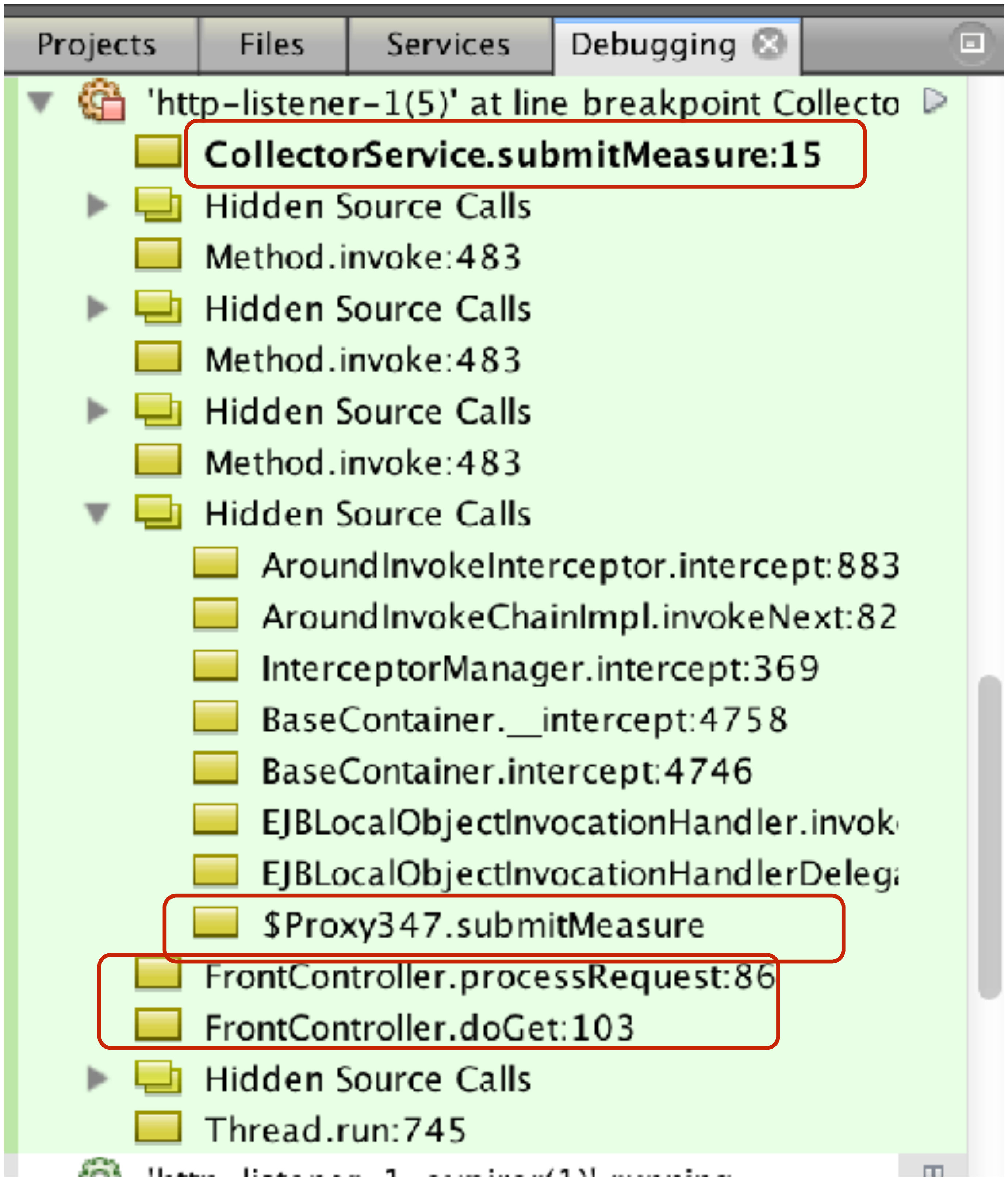
“How is the container able to **intercept** method calls on EJBs?”

“I believe that it generates **dynamic proxies**”

“**If** I am right, then I should see these proxies in the call stack.”

“Let’s **inspect the call stack in the debugger**”

“In the debugger console, I can confirm that there are container generated classes between my servlet and my service implementation.”



```
'http-listener-1(5)' at line breakpoint Collecto
  CollectorService.submitMeasure:15
  Hidden Source Calls
  Method.invoke:483
  Hidden Source Calls
  Method.invoke:483
  Hidden Source Calls
  Method.invoke:483
  Hidden Source Calls
    AroundInvokeInterceptor.intercept:883
    AroundInvokeChainImpl.invokeNext:82
    InterceptorManager.intercept:369
    BaseContainer.__intercept:4758
    BaseContainer.intercept:4746
    EJBLocalObjectInvocationHandler.invok
    EJBLocalObjectInvocationHandlerDeleg
    $Proxy347.submitMeasure
  FrontController.processRequest:86
  FrontController.doGet:103
  Hidden Source Calls
  Thread.run:745
```

At some point, the method call is forwarded to my implementation.

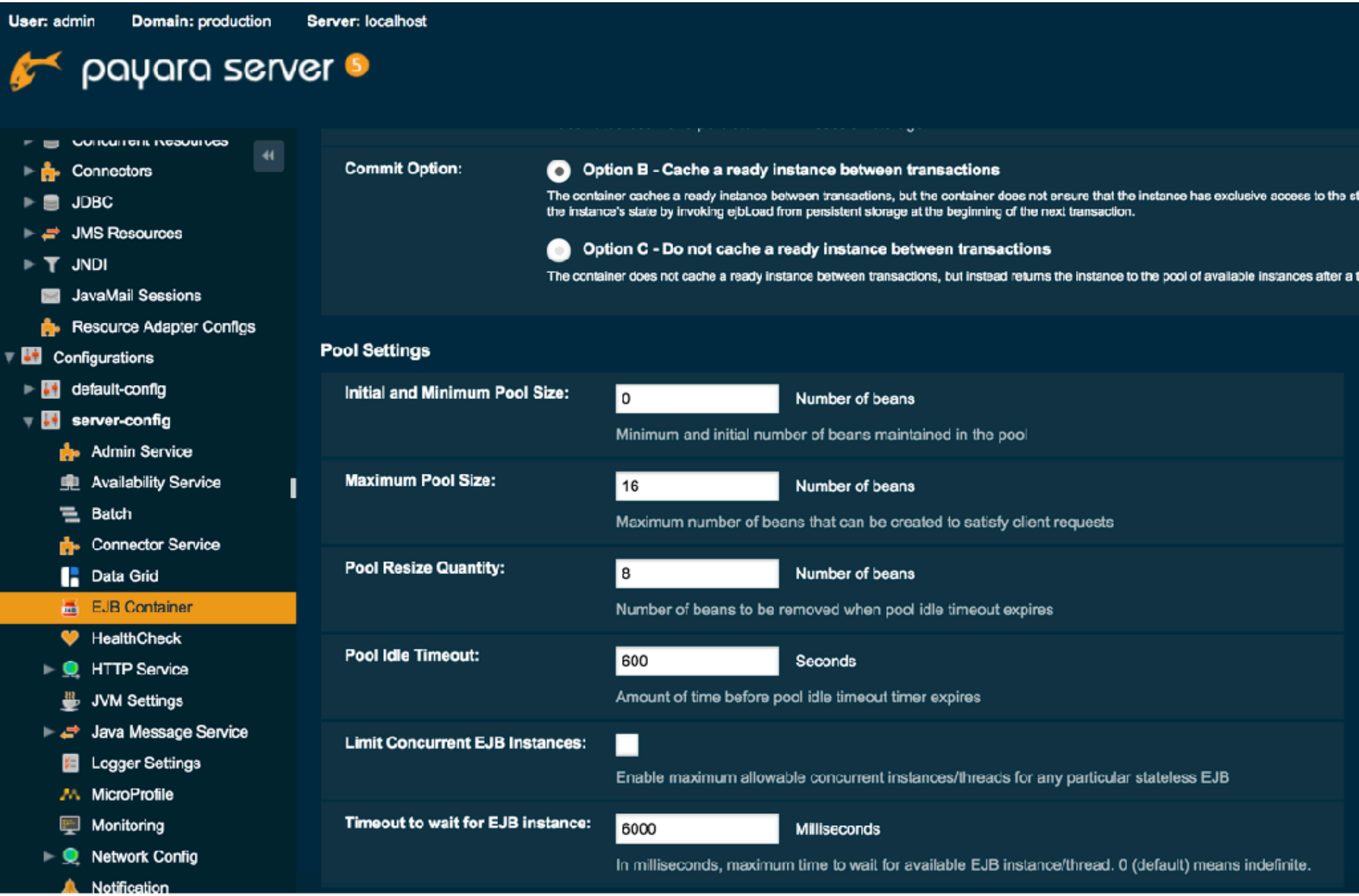
The reference actually points to a proxy generated by the container. The container performs tasks that are visible in a **long call stack**!

My servlet invokes the method on its **reference** to the EJB.

*An HTTP request has arrived; GF invokes the doGet callback on my servlet (**IoC**). GF has also **injected** a **reference** to the EJB into the servlet.*

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

The book talks about pooling... what does it mean and why is it useful?


User: admin Domain: production Server: localhost

Commit Option: [✓ Option B - Cache a ready instance between transactions]

The container caches a ready instance between transactions, but the container does not ensure that the instance has exclusive access to the st the instance's state by invoking ejbLoad from persistent storage at the beginning of the next transaction.

Option C - Do not cache a ready instance between transactions

The container does not cache a ready instance between transactions, but instead returns the instance to the pool of available instances after a t

Pool Settings

Initial and Minimum Pool Size:	0	Number of beans
	Minimum and initial number of beans maintained in the pool	
Maximum Pool Size:	16	Number of beans
	Maximum number of beans that can be created to satisfy client requests	
Pool Resize Quantity:	8	Number of beans
	Number of beans to be removed when pool idle timeout expires	
Pool Idle Timeout:	600	Seconds
	Amount of time before pool idle timeout timer expires	
Limit Concurrent EJB Instances:	☐	
	Enable maximum allowable concurrent instances/threads for any particular stateless EJB	
Timeout to wait for EJB instance:	6000	Milliseconds
	In milliseconds, maximum time to wait for available EJB instance/thread. 0 (default) means indefinite.	

Why pool objects?

There are 2 main reasons for pooling objects

- To increase performance. Some objects take a long time to be created and initialized (e.g. DB connection object). It's better to reuse objects instead of throwing them away and recreating them.
- To set a limit on resource consumption (CPU, RAM). Under heavy load, we decide how many requests we process at the same time. It's better to have clients wait bit than to exhaust all server resources.

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Introduction to JMeter

JMeter

- Open source project, apache foundation
- http://jmeter.apache.org/index.html

“The Apache JMeter™ desktop application is open source software, a 100% pure Java application designed to ***load test functional behavior*** *and* ***measure performance****.*
It was originally designed for ***testing Web Applications*** *but has since* ***expanded to other*** *test functions.”*

JMeter

"Apache JMeter may be used to ***test performance*** *both on static and dynamic resources (files, Servlets, Perl scripts, Java Objects, Data Bases and Queries, FTP Servers and more).*

It can be used to ***simulate a heavy load*** *on a server, network or object to test its strength or to analyze overall performance under* ***different load types****. You can use it to make a* ***graphical analysis of performance*** *or to test your server/script/object behavior under heavy* ***concurrent*** *load."*

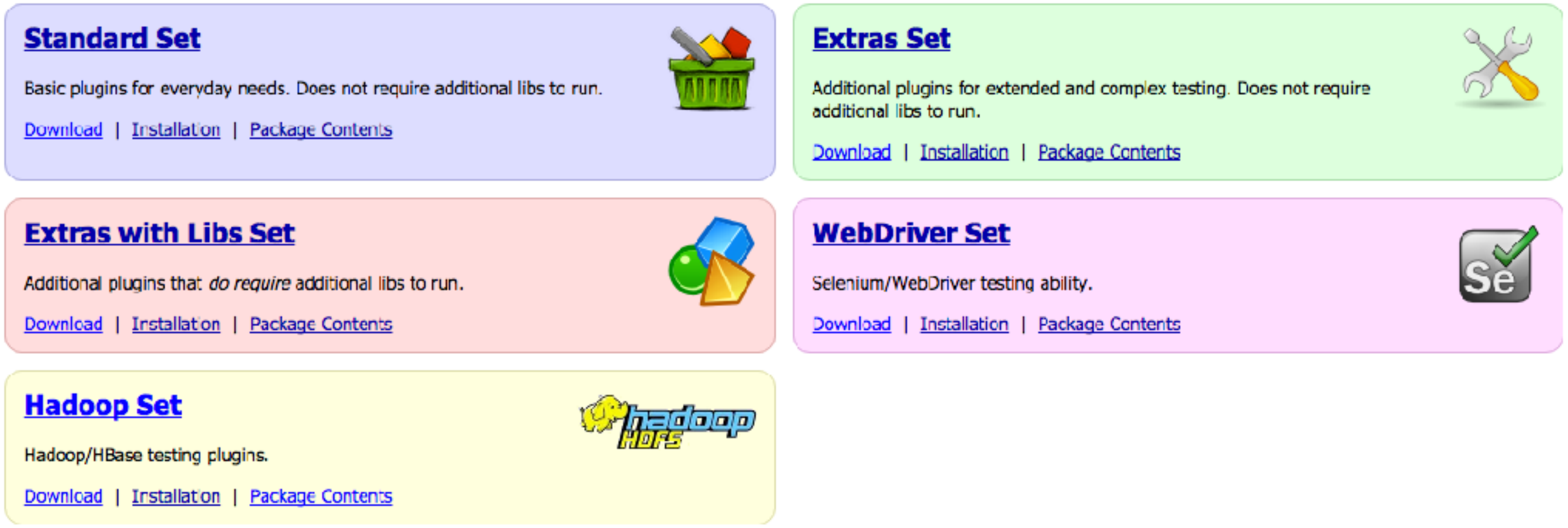
Types of tests (1)

- **Functional tests**
  - Is the system doing what it is supposed to do?
  - Does its behavior comply with functional requirements (use cases)?
  - Selenium is a tool for automating functional testing of web applications (http://seleniumhq.org/)
- **Performance, load and stress tests**
  - What is the response time? What is the consumption of resources? Are there issues (e.g. concurrency issues) that happen under load?
  - Relevant both for interactive and batch use cases.

What to install?

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

- **Main project**
  - http://jmeter.apache.org/download_jmeter.cgi
- **Add-ons**
  - http://jmeter-plugins.org/



Standard Set

Basic plugins for everyday needs. Does not require additional libs to run.

Download | Installation | Package Contents

Extras Set

Additional plugins for extended and complex testing. Does not require additional libs to run.

Download | Installation | Package Contents

Extras with Libs Set

Additional plugins that *do require* additional libs to run.

Download | Installation | Package Contents

WebDriver Set

Selenium/WebDriver testing ability.

Download | Installation | Package Contents

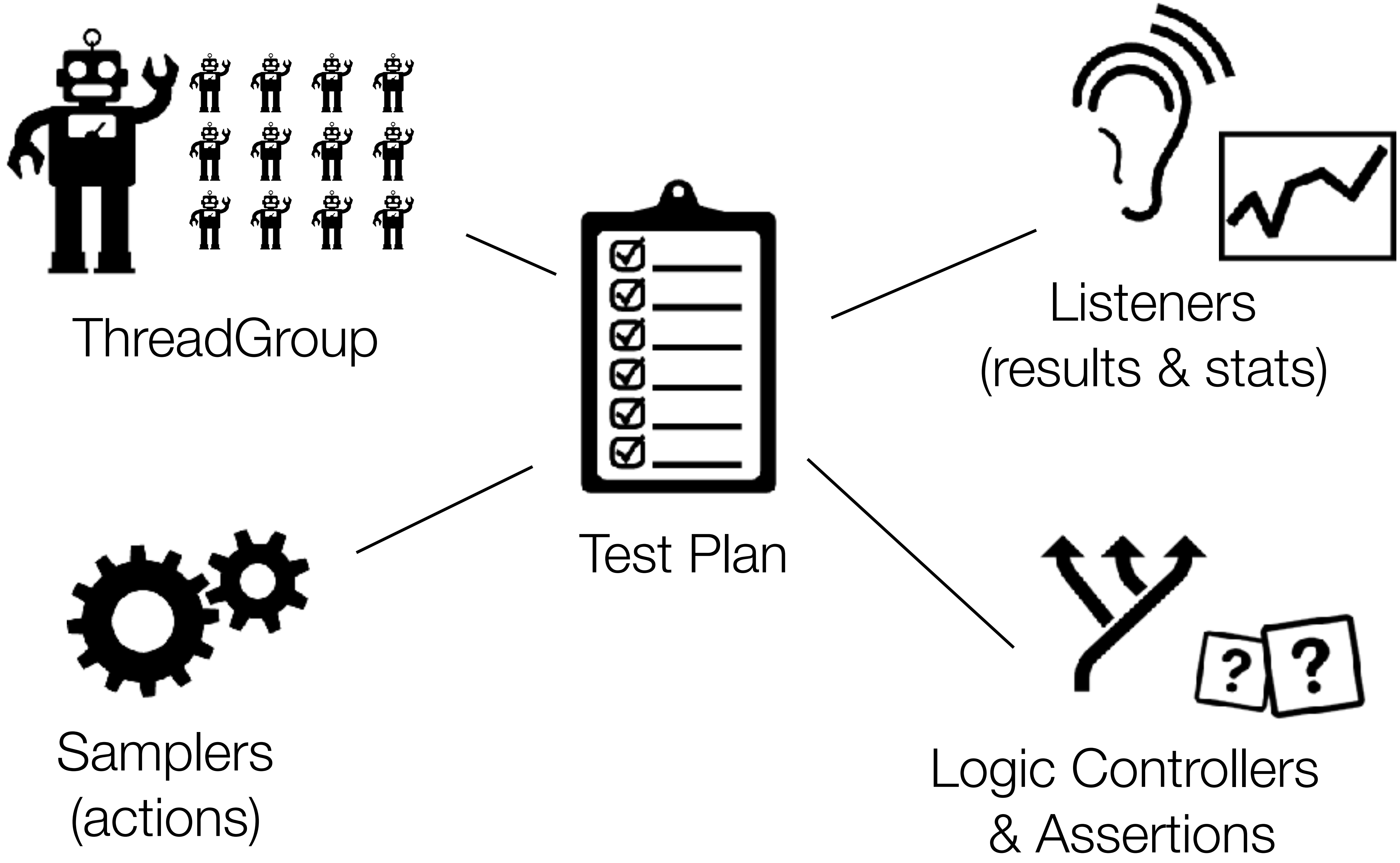
Hadoop Set

Hadoop/HBase testing plugins.

Download | Installation | Package Contents

JMeter Building Blocks

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud



Concepts

- Test Plan
- ThreadGroup
- Samplers
- Logic Controllers
- Listeners
- Timers
- Assertions
- Configuration Elements
- Pre-Processor Elements
- Post-Processor Elements

Thread Group

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

*"Thread group elements are the **beginning points of any test plan**. All controllers and samplers must be under a thread group. [...]. As the name implies, the thread group element controls the number of threads JMeter will use to execute your test. The controls for a thread group allow you to:*

- *Set the **number of threads***
- *Set the **ramp-up** period*
- *Set the **number of times** to execute the test*

*Each thread will execute the test plan in its entirety and completely independently of other test threads. **Multiple threads are used to simulate concurrent connections to your server application.**"*

Sampler

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

"Samplers tell JMeter to ***send requests to a server and wait for a response****. They are processed in the order they appear in the tree. Controllers can be used to modify the number of repetitions of a sampler.*

- *FTP Request*
- ***HTTP Request***
- *JDBC Request*
- *Java object request*
- *LDAP Request*
- *SOAP/XML-RPC Request*
- *WebService (SOAP) Request*

Each sampler has several ***properties*** *you can set. You can further customize a sampler by adding one or more Configuration Elements to the Test Plan."*

Logic Controllers

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

"Logic Controllers let you customize ***the logic that JMeter uses to decide when to send requests****. Logic Controllers can change the order of requests coming from their child elements. They can modify the requests themselves, cause JMeter to repeat requests, etc."*

- *Loop Controller*
- *Once Only Controller*
- *Interleave Controller*
- *Random Controller*
- *Random Order Controller*
- *Throughput Controller*
- *Runtime Controller*
- *If Controller*
- *etc.*

Listeners

*“Listeners provide **access to the information JMeter gathers about the test cases** while JMeter runs. The **Graph Results** listener plots the response times on a graph. The "**View Results Tree**" Listener shows details of sampler requests and responses, and can display basic HTML and XML representations of the response. Other listeners provide **summary or aggregation information**.*

*Additionally, listeners can **direct the data to a file** for later use.*

Listeners can be added anywhere in the test, including directly under the test plan. They will collect data only from elements at or below their level.”

Timers

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

"By default, a JMeter thread sends requests without pausing between each request. We recommend that you specify a delay by adding one of the available timers to your Thread Group. If you do not add a delay, JMeter could ***overwhelm your server*** *by making too many requests in a very short amount of time.*
The timer will cause JMeter to delay a certain amount of time before each sampler which is in its scope .

If you choose to add more than one timer to a Thread Group, JMeter takes the sum of the timers and pauses for that amount of time before executing the samplers to which the timers apply. Timers can be added as children of samplers or controllers in order to restrict the samplers to which they are applied.

To provide a pause at a single place in a test plan, one can use the ***Test Action Sampler.****"*

Assertions

“Assertions allow you to ***assert facts about responses received*** *from the server being tested.*

Using an assertion, you can essentially ***"test" that your application is returning the results you expect it to****.*

For instance, you can assert that the response to a query will ***contain some particular text****. The text you specify can be a Perl-style regular expression, and you can indicate that the response is to contain the text, or that it should match the whole response.*

You can add an assertion to any Sampler. For example, you can add an assertion to a HTTP Request that checks for the text, "</HTML>". JMeter will then check that the text is present in the HTTP response. If JMeter cannot find the text, then it will ***mark this as a failed request.****”*

Configuration Elements

“A configuration element works closely with a Sampler. Although it does not send requests (except for HTTP Proxy Server), it can add to or modify requests.

*A configuration element is accessible from only inside the tree branch where you place the element. For example, if you place an **HTTP Cookie Manager** inside a Simple Logic Controller, the Cookie Manager will only be accessible to HTTP Request Controllers you place inside the Simple Logic Controller”*

- *HTTP Authorization Manager*
- *HTTP Cache Manager*
- *HTTP Cookie Manager*
- *HTTP Request Defaults*
- *HTTP Header Manager*

How to Create Test Scenarios?

- **Option 1 : manually**
  - Create a Test Plan
  - Add a Thread Group
  - Add HTTP samplers and specify HTTP request parameters
- **Option 2 : recording with JMeter configured as an HTTP proxy**
  - http://jmeter.apache.org/usermanual/jmeter_proxy_step_by_step.pdf
  - Do manual adjustments

Advanced Usage

- **Use variables:**
  - Very often, it is needed to use parts of an HTTP response into follow-up requests (e.g. session ids).
  - http://jmeter.apache.org/usermanual/test_plan.html#properties
- **Use more than one machine:**
  - With JMeter running on a single machine, it is common to “exhaust” the client before the server (especially when testing a “real” infrastructure with multiple nodes).
  - For real performance tests, it is therefore recommended to use multiple machines for injecting load into the network. One way to do it is use use virtual machines in a cloud environment (e.g. on Amazon EC2).
  - Multiple JMeter clients can be coordinated by a master and results can be collected and aggregated (http://jmeter.apache.org/usermanual/jmeter_distributed_testing_step_by_step.pdf)

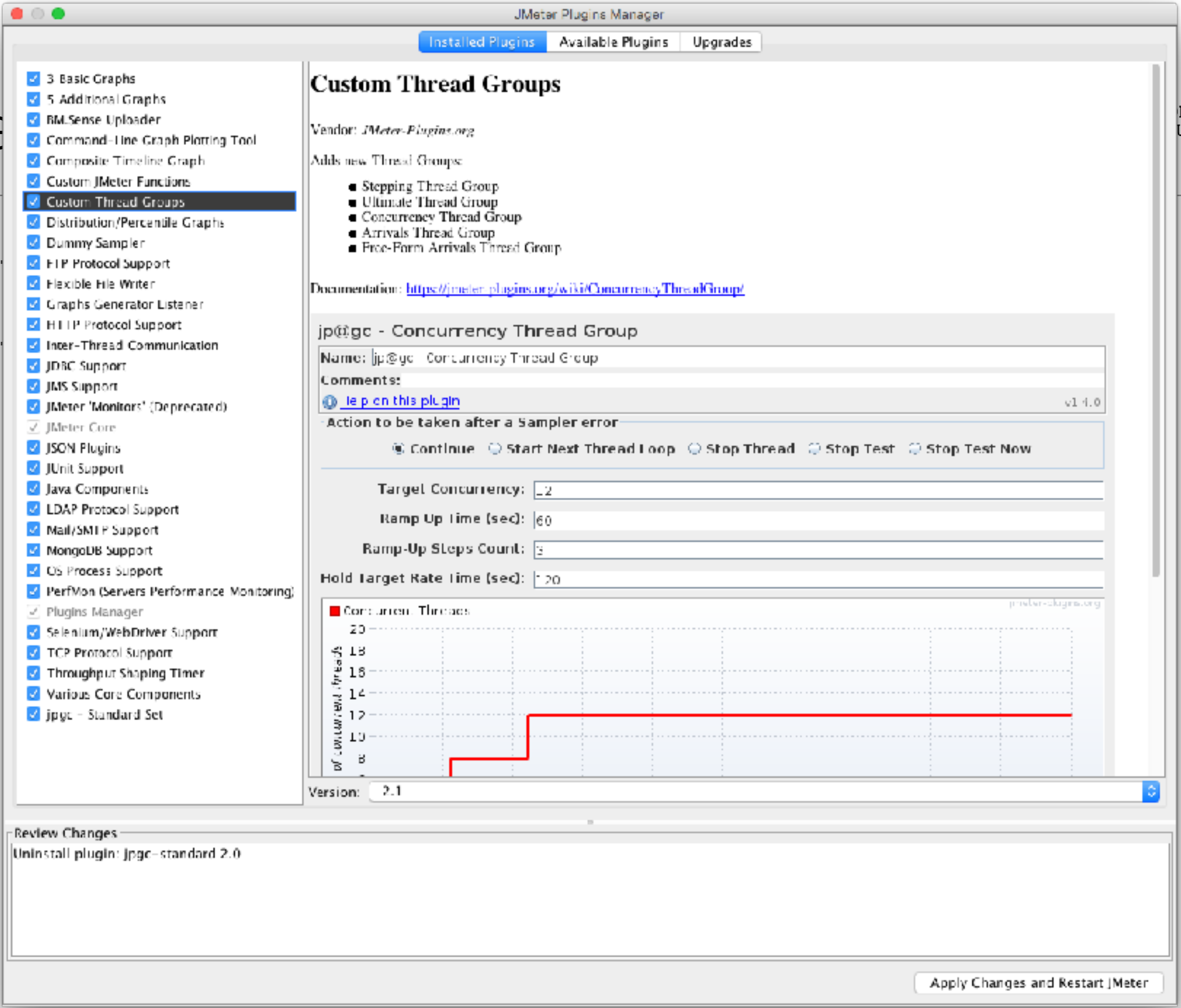
To run the example

- **Install JMeter**
  - http://jmeter.apache.org/
- **Install the JMeter plugin manager**
  - https://jmeter-plugins.org/install/Install/
- Create a test plan, with at the minimum:
  - An **ultimate thread group** (plugin to install with the plugin manager)
  - An **HTTP request sampler**
  - A **constant timer**
  - **Listeners** to display the results (play with “summary report”, “response time graph”, “view results in tree”).

Exe

E GESTION
UD

- Ins
- Ins
-



JMeter Plugins Manager

Installed Plugins | Available Plugins | Upgrades

- ✓ 3 Basic Graphs
- ✓ 5 Additional Graphs
- ✓ BM.Sense Uploader
- ✓ Command-Line Graph Plotting Tool
- ✓ Composite Timeline Graph
- ✓ Custom JMeter Functions
- ✓ Custom Thread Groups
- ✓ Distribution/Percentile Graphs
- ✓ Dummy Sampler
- ✓ FTP Protocol Support
- ✓ Flexible File Writer
- ✓ Graphs Generator Listener
- ✓ HTTP Protocol Support
- ✓ Inter-Thread Communication
- ✓ JDBC Support
- ✓ JMS Support
- ✓ JMeter 'Monitors' (Deprecated)
- ✓ JMeter Core
- ✓ JSON Plugins
- ✓ JUnit Support
- ✓ Java Components
- ✓ LDAP Protocol Support
- ✓ Mail/SMTP Support
- ✓ MongoDB Support
- ✓ OS Process Support
- ✓ PerfMon (Servers Performance Monitoring)
- ✓ Plugins Manager
- ✓ Selenium/WebDriver Support
- ✓ TCP Protocol Support
- ✓ Throughput Shaping Timer
- ✓ Various Core Components
- ✓ jpgc - Standard Set

Custom Thread Groups

Vendor: *JMeter-Plugins.org*

Adds new Thread Groups:

- Stepping Thread Group
- Ultimate Thread Group
- Concurrency Thread Group
- Arrivals Thread Group
- Free-Form Arrivals Thread Group

Documentation: https://jmeter-plugins.org/wiki/ConcurrencyThreadGroup/

jp@gc - Concurrency Thread Group

Name: jp@gc - Concurrency Thread Group

Comments:

Help on this plugin — v1.4.0

Action to be taken after a Sampler error

[✓ Continue] Start Next Thread Loop | Stop Thread | Stop Test | Stop Test Now

Target Concurrency: 12

Ramp Up Time (sec): 60

Ramp-Up Steps Count: 3

Hold Target Rate Time (sec): 120

Version: 2.1

Review Changes

Uninstall plugin: jpgc-standard 2.0

Apply Changes and Restart JMeter

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Experiment with JMeter

Challenge

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Design an experiment to:

- prove that the application server manages pools of Stateless Session Beans (multiple instances)
- measure how the size of the pool impacts the throughput of the application
- measure how the size of the pool impacts the resource consumption (RAM, CPU)

Hints

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

You should use a combination of tools

- JMeter to generate the load
- VisualVM (or JConsole) to monitor resource consumption on the server (container) and client (jmeter) side.
- You can use tricks in the code to simulate a time consuming task (Thread.sleep), or a resource hungry task (allocate dummy objects).