



<XML num="4">Langage Web 2</XML>

- Emilien Bondu (Airbus Defence and Space)
- emilien.bondu@cassidian.com



Sommaire

- Cours 1 : Introduction à XML et validation (XSD)
- Cours 2 : Validation (DTD, Relax NG, Schematron) et transformation (XPath, XSL; XSL-FO)
- Cours 3 : Recherche XML (XQuery), Base de données XML, Liens XML, Manipulation XML en Java: Dom, Sax
- Cours 4 : Manipulation XML en Java : StAX, Data-binding, JavaEE : tomcat
- Cours 5 : Manipulation XML en JavaEE: servlet, JSP, ExpressionLanguage, TagLib
- Cours 6 : TagLib (suite), Spring, JSF, AJAX



Rappels

- XPointer permet :
 - de pointer une partie d'un fichier XML
 - de réaliser un pointeur mémoire vers un fichier XML
 - de désigner des balises d'un arbre XML
 - de désigner des parties de contenu au sein d'un arbre XML



Rappels

- XLink permet :
 - de pointer une partie d'un fichier XML
 - de créer des liens multidirectionnel entre fichiers XML
 - de préciser le comportement des applications par rapport aux liens
 - d'associer de la sémantique aux liens et aux ressources



Rappels

- XQuery permet :
 - de produire du XML
 - de rechercher des documents XML
 - de rechercher des parties de documents XML
 - d'échanger des données entre applications



Rappels

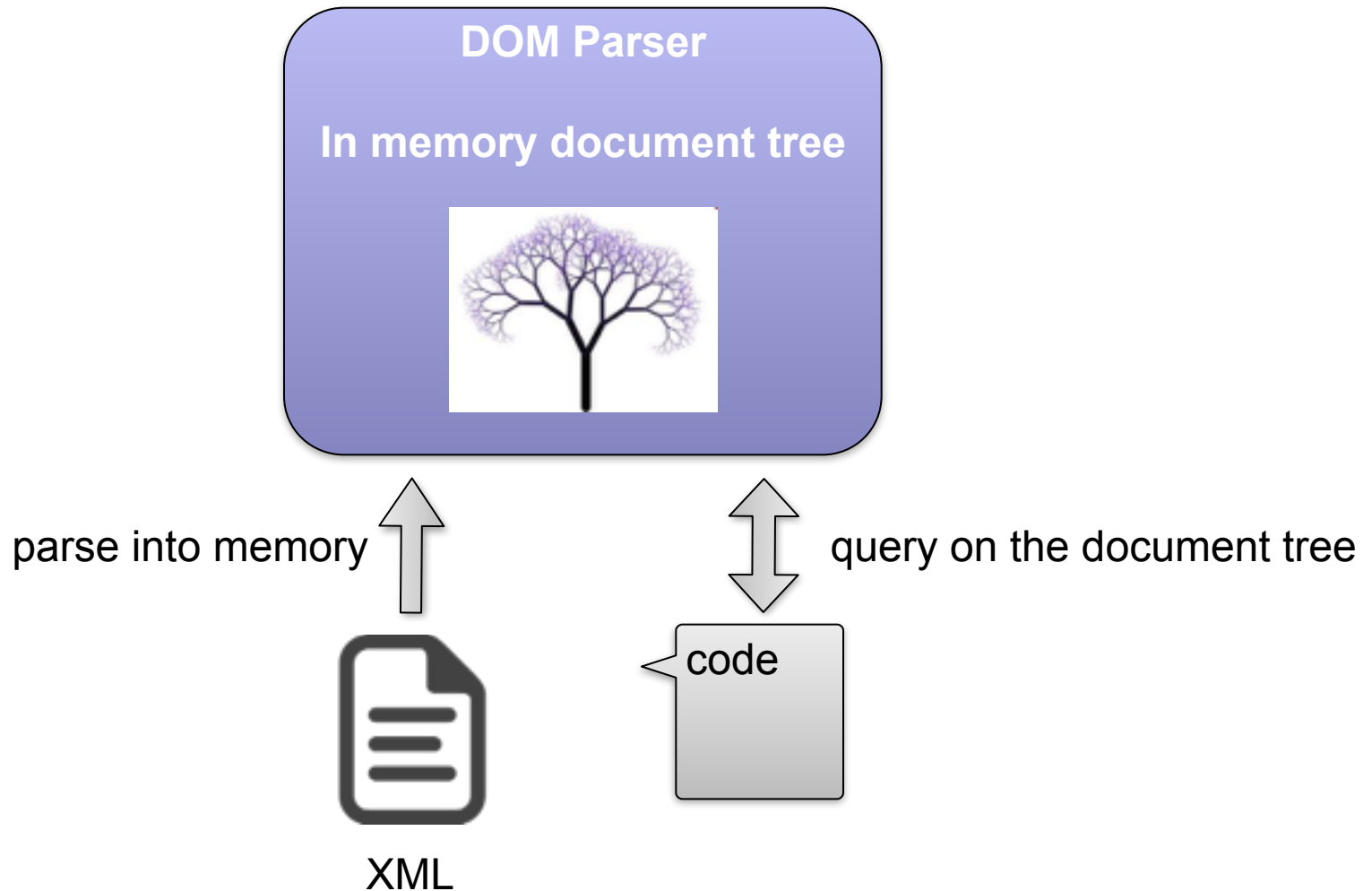
- XQuery permet :
 - de faire de la recherche plein texte
 - d'ajouter/supprimer des données
 - de définir des variables
 - d'écrire des requêtes avec une structure FLWOR



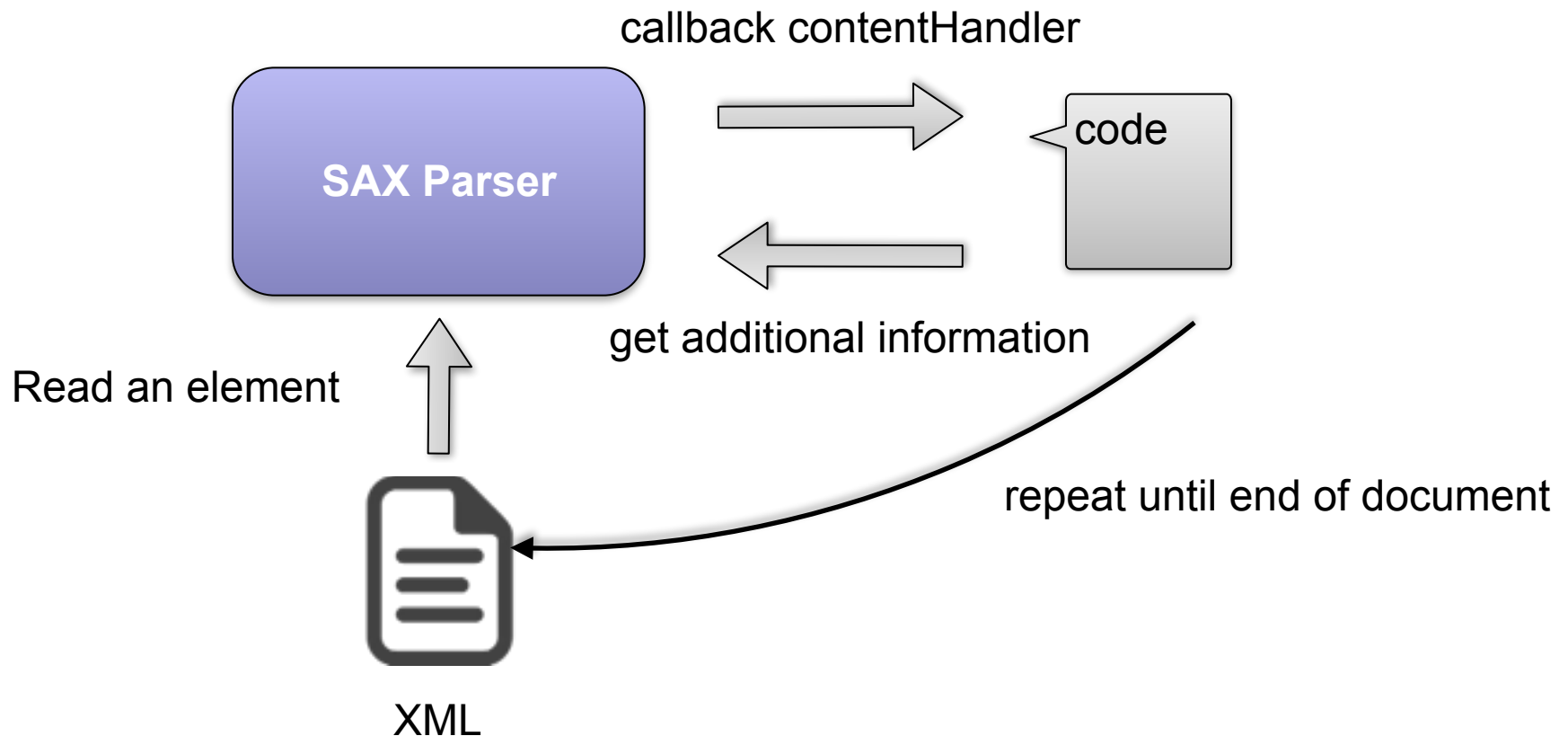
Rappels

- Une BD XML Native :
 - permet de stocker des fichiers XML
 - peut offrir des APIs de manipulations
 - doit offrir une interface XQuery
 - doit stocker ces fichiers en XML

Rappels DOM



Rappels SAX





StAX

- **Streaming API for XML**
 - Combiner les avantages de DOM et SAX
 - Lecture et écriture dans un flux XML
 - Itérer sur les événements du flux

StAX

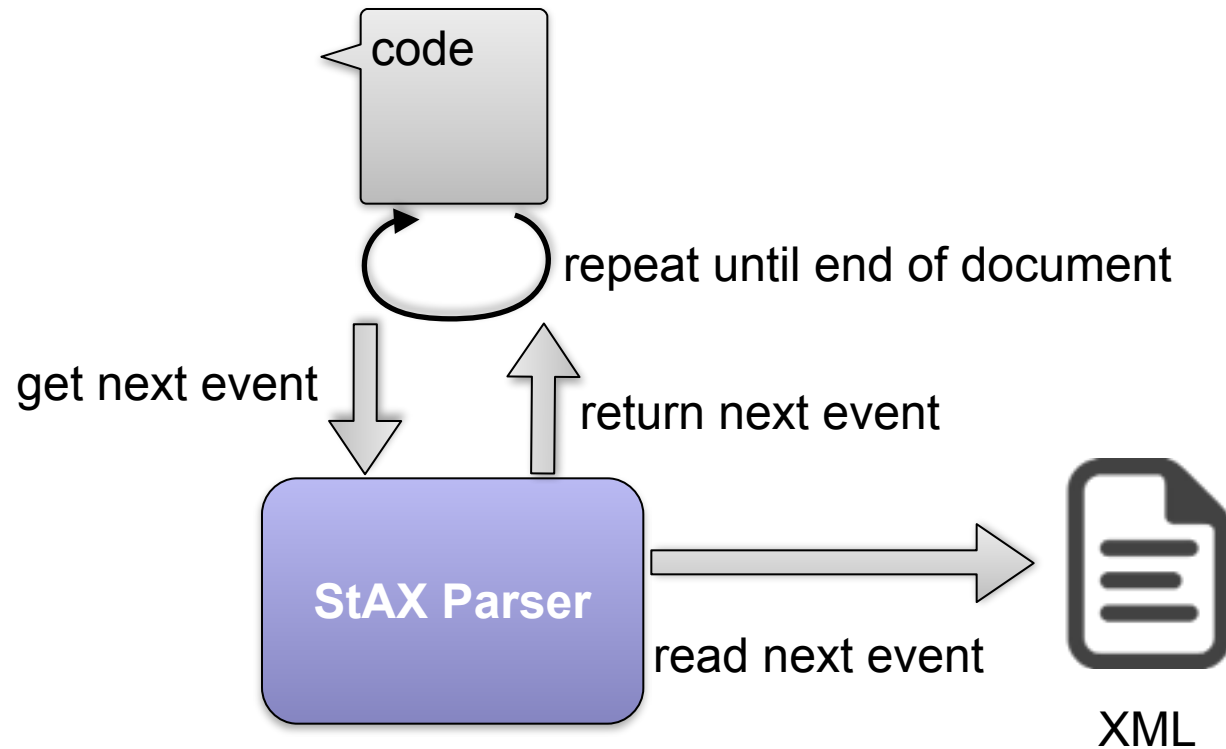
- **Avantages**

- ☐ Faible utilisation mémoire
- ☐ Ecriture possible

- **Limites**

- ☐ Pas de retour en arrière
- ☐ Construction séquentielle
- ☐ Pas de validation

StAX : fonctionnement





StAX

- **Implémentations (JSR 173)**
 - Sun Java StAX XML Processor
 - Codehaus Stax
 - Woodstox
 - Aalto

StAX : Cursor API



- **XMLInputFactory** (oracle): Entrée sur un flux XML
 - `newInstance()`, `createXMLStreamReader()`, `createXMLStreamReader()`
 - **XMLStreamReader** : Lecture du flux
 - `hasNext()`, `next()`, `nextTag()`, `getEventType()`, `getNamespaceURI()`, `getLocalName()`, `getText()`, `getAttributeType(int index)`, `getAttributeValue(String namespaceUri, String localName)`, `isAttributeSpecified(int index)`
-
- **XMLOutputFactory** : Sortie sur un flux XML
 - `newInstance()`, `createXMLStreamWriter()`, `createXMLStreamWriter()`
 - **XMLStreamWriter** : Ecriture dans le flux
 - `writeStartDocument()`, `writeEndDocument()`, `writeStartElement(String localname)`, `writeEndElement()`, `writeAttribute()`, `writeCharacters()`, `setPrefix()`, `setDefaultNamespace()`...

StAX : Cursor API



<i>type event</i>	<i>methodes valides</i>
all	getProperty(), hasNext(), require(), close(), getNamespaceURI(), isStartElement(), isEndElement(), isCharacters(), isWhiteSpace(), getNamespaceContext(), getEventType(), getLocation(), hasText(), hasName()
START_ELEMENT	next(), getName(), getLocalName(), hasName(), getPrefix(), getAttributeXXX(), isAttributeSpecified(), getNamespaceXXX(), getElementText(), nextTag()
ATTRIBUTE	next(), nextTag() getAttributeXXX(), isAttributeSpecified()
NAMESPACE	next(), nextTag() getNamespaceXXX()
END_ELEMENT	next(), getName(), getLocalName(), hasName(), getPrefix(), getNamespaceXXX(), nextTag()
CHARACTERS	next(), getTextXXX(), nextTag()
CDATA	next(), getTextXXX(), nextTag()
COMMENT	next(), getTextXXX(), nextTag()
SPACE	next(), getTextXXX(), nextTag()
START_DOCUMENT	next(), getEncoding(), getVersion(), isStandalone(), standaloneSet(), getCharacterEncodingScheme(), nextTag()
END_DOCUMENT	close()
PROCESSING_INSTRUCTION	next(), getPITarget(), getPIData(), nextTag()
ENTITY_REFERENCE	next(), getLocalName(), getText(), nextTag()
DTD	next(), getText(), nextTag()

StAX : Iterator API



- **XMLInputFactory** (oracle): Entrée sur un flux XML
 - `newInstance()`, `createXMLStreamReader()`, `createXMLEventReader()`
- **XMLEventReader** : Lecture événementielle du flux
 - `hasNext()`, `nextEvent()`, `nextTag()`, `peek()`, `getElementText()` ...
- **XMLEvent** : événement
 - `isStartElement()`, `isStartDocument()`, `isEndElement()`, `isCharacters()`, `isAttribute()`, `isNamespace()`, `isEntityReference()` ...

- **XMLOutputFactory** : Sortie sur un flux XML
 - `newInstance()`, `createXMLStreamWriter()`, `createXMLEventWriter()`
- **XMLEventFactory** : création des événements
 - `createElement()`, `createAttribute()`, `createDocument()`, `createEntity()`...
- **XMLEventWriter** : Ecriture dans le flux
 - `add(XMLEvent event)`, `add(XMLEventReader reader)`, `close()`, `setPrefix()`, `setDefaultNamespace()` ...



StAX : exemple

```
public void testCursor() throws FileNotFoundException, XMLStreamException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    File file = new File("src/livres/livre.xml");
    XMLStreamReader reader = factory.createXMLStreamReader(new FileReader(file));

    while (reader.hasNext()) {
        int type = reader.next();

        switch (type) {
            case XMLStreamReader.START_ELEMENT:
                System.out.println("debut - "+reader.getLocalName());

            case XMLStreamReader.END_ELEMENT:
                System.out.println("fin - "+reader.getLocalName());
        }
    }
}
```

StAX : exemple

```
public void testCursor() throws FileNotFoundException, XMLStreamException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    File file = new File("src/livres/livre.xml");
    XMLStreamReader reader = factory.createXMLStreamReader(new FileReader(file));

    while (reader.hasNext()) {
        int type = reader.next();

        switch (type) {
            case XMLStreamReader.START_ELEMENT:
                System.out.println("debut - "+reader.getLocalName());

            case XMLStreamReader.END_ELEMENT:
                System.out.println("fin - "+reader.getLocalName());
        }
    }
}
```

debut - livre
fin - livre
debut - titre
fin - titre
debut - auteur
fin - auteur
debut - nom
fin - nom
fin - nom
debut - prenom
fin - prenom
fin - prenom
fin - auteur
debut - chapitre
fin - chapitre
debut - titre
fin - titre
fin - titre
debut - paragraphe
fin - paragraphe
debut - titre
fin - titre
fin - titre
debut - corps
fin - corps
fin - corps
fin - paragraphe
debut - paragraphe
fin - paragraphe
debut - titre
fin - titre
fin - titre
debut - corps
fin - corps
fin - corps
fin - paragraphe
fin - chapitre
debut - chapitre
fin - chapitre
debut - titre
fin - titre
fin - titre
debut - paragraphe
fin - paragraphe
debut - titre
fin - titre
fin - titre
debut - corps
fin - corps
fin - corps
fin - paragraphe
fin - chapitre
fin - livre



StAX : exemple

```
public void testIterator() throws FileNotFoundException, XMLStreamException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    File file = new File("src/livres/livre.xml");
    XMLEventReader reader = factory.createXMLEventReader(new FileReader(file));

    while (reader.hasNext()) {
        XMLEvent next = reader.nextEvent();

        if(next.isStartElement()){
            StartElement start = next.asStartElement();
            System.out.println("debut - "+start.getName().getLocalPart());
        }
        if(next.isEndElement()){
            EndElement end = next.asEndElement();
            System.out.println("fin - "+end.getName().getLocalPart());
        }
    }
}
```



StAX : exemple

```
public void testIterator() throws FileNotFoundException, XMLStreamException {  
  
    XMLInputFactory factory = XMLInputFactory.newInstance();  
    File file = new File("src/livres/livre.xml");  
    XMLStreamReader reader = factory.createXMLStreamReader(new FileReader(file));  
  
    while (reader.hasNext()) {  
        XMLEvent next = reader.nextEvent();  
  
        if(next.isStartElement()){  
            StartElement start = next.asStartElement();  
            System.out.println("debut - "+start.getName().getLocalPart());  
        }  
        if(next.isEndElement()){  
            EndElement end = next.asEndElement();  
            System.out.println("fin - "+end.getName().getLocalPart());  
        }  
    }  
}
```

```
debut - livre  
debut - titre  
fin - titre  
debut - auteur  
debut - nom  
fin - nom  
debut - prenom  
fin - prenom  
fin - auteur  
debut - chapitre  
debut - titre  
fin - titre  
debut - paragraphe  
debut - titre  
fin - titre  
debut - corps  
fin - corps  
fin - paragraphe  
debut - paragraphe  
debut - titre  
fin - titre  
debut - corps  
fin - corps  
fin - paragraphe  
fin - chapitre  
debut - chapitre  
debut - titre  
fin - titre  
debut - paragraphe  
debut - titre  
fin - titre  
debut - corps  
fin - corps  
fin - paragraphe  
fin - chapitre  
fin - livre
```

StAX : exemple

```
public void testIteratorWrite() throws XMLStreamException, IOException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    XMLOutputFactory factoryOut = XMLOutputFactory.newInstance();

    File file = new File("src/livres/livre.xml");
    File out = new File("src/livres/out.xml");

    XMLEventReader reader = factory.createXMLEventReader(new FileReader(file));
    XMLEventWriter writer = factoryOut.createXMLEventWriter(new FileWriter(out));

    XMLEventFactory eventFactory = XMLEventFactory.newInstance();

    writer.add(eventFactory.createStartDocument());
    writer.add(
        eventFactory.createStartElement("livre", "http://livre.org", "newLivre"));

    while (reader.hasNext()) {
        XMLEvent next = reader.nextEvent();

        if (next.isStartElement()) {
            writer.add(next);
        }
    }

    writer.add(eventFactory.createEndDocument());
    writer.close();
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<livre xmlns="http://livre.org"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://livre.org livre.xsd">

    <titre>Apprendre le XML</titre>
    <auteur>
        <nom>Bondu</nom>
        <prenom>Emilien</prenom>
    </auteur>
    <!-- Premier chapitre du livre -->
    <chapitre num="1">
        <titre>Les Bases d'XML</titre>
        <paragraphe>
            <titre>Premier paragraphe</titre>
            <corps>Contenu du premier paragraphe...</corps>
        </paragraphe>
        <paragraphe>
            <titre>Deuxième paragraphe</titre>
            <corps>Contenu du deuxième paragraphe...</corps>
        </paragraphe>
    </chapitre>
    <!-- Premier chapitre du livre -->
    <chapitre num="2">
        <titre>La transformation XML</titre>
        <paragraphe>
            <titre>Premier paragraphe</titre>
            <corps>Contenu du premier paragraphe...</corps>
        </paragraphe>
    </chapitre>
</livre>
```

StAX : exemple

```
public void testIteratorWrite() throws XMLStreamException, IOException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    XMLOutputFactory factoryOut = XMLOutputFactory.newInstance();

    File file = new File("src/livres/livre.xml");
    File out = new File("src/livres/out.xml");

    XMLEventReader reader = factory.createXMLEventReader(new FileReader(file));
    XMLEventWriter writer = factoryOut.createXMLEventWriter(new FileWriter(out));

    XMLEventFactory eventFactory = XMLEventFactory.newInstance();

    writer.add(eventFactory.createStartDocument());
    writer.add(
        eventFactory.createStartElement("livre", "http://livre.org", "newLivre"));

    while (reader.hasNext()) {
        XMLEvent next = reader.nextEvent();

        if (next.isStartElement()) {
            writer.add(next);
        }
    }

    writer.add(eventFactory.createEndDocument());
    writer.close();
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<livre:newLivre>
  <livre xmlns="http://livre.org"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://livre.org livre.xsd">
    <titre>
      <auteur>
        <nom>
          <prenom>
            <chapitre num="1">
              <titre>
                <paragraphe>
                  <titre>
                    <corps>
                      <paragraphe>
                        <titre>
                          <corps>
                            <chapitre num="2">
                              <titre>
                                <paragraphe>
                                  <titre>
                                    <corps></corps>
                                  </titre>
                                </paragraphe>
                              </titre>
                            </chapitre>
                          </corps>
                        </titre>
                      </paragraphe>
                    </corps>
                  </titre>
                </paragraphe>
              </titre>
            </chapitre>
          </prenom>
        </nom>
      </auteur>
    </titre>
  </livre>
</livre:newLivre>
```

StAX : exemple

```
@Test
public void testIteratorWrite() throws XMLStreamException, IOException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    XMLOutputFactory factoryOut = XMLOutputFactory.newInstance();

    File file = new File("src/livres/livre.xml");
    File out = new File("src/livres/out.xml");

    XMLEventReader reader = factory.createXMLEventReader(new FileReader(file));
    XMLEventWriter writer = factoryOut.createXMLEventWriter(new FileWriter(out));

    XMLEventFactory eventFactory = XMLEventFactory.newInstance();

    writer.add(eventFactory.createStartDocument());
    writer.add(
        eventFactory.createStartElement("livre", "http://livre.org", "newLivre"));

    while (reader.hasNext()) {
        XMLEvent next = reader.nextEvent();

        if (next.isStartElement()) {
            writer.add(next);
        } else if (next.isEndElement()) {
            writer.add(next);
        }
    }

    writer.add(eventFactory.createEndDocument());
    writer.close();
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<livre xmlns="http://livre.org"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://livre.org livre.xsd">

    <titre>Apprendre le XML</titre>
    <auteur>
        <nom>Bondu</nom>
        <prenom>Emilien</prenom>
    </auteur>
    <!-- Premier chapitre du livre -->
    <chapitre num="1">
        <titre>Les Bases d'XML</titre>
        <paragraphe>
            <titre>Premier paragraphe</titre>
            <corps>Contenu du premier paragraphe...</corps>
        </paragraphe>
        <paragraphe>
            <titre>Deuxième paragraphe</titre>
            <corps>Contenu du deuxième paragraphe...</corps>
        </paragraphe>
    </chapitre>
    <!-- Premier chapitre du livre -->
    <chapitre num="2">
        <titre>La transformation XML</titre>
        <paragraphe>
            <titre>Premier paragraphe</titre>
            <corps>Contenu du premier paragraphe...</corps>
        </paragraphe>
    </chapitre>
</livre>
```

StAX : exemple

```
@Test
public void testIteratorWrite() throws XMLStreamException, IOException {

    XMLInputFactory factory = XMLInputFactory.newInstance();
    XMLOutputFactory factoryOut = XMLOutputFactory.newInstance();

    File file = new File("src/livres/livre.xml");
    File out = new File("src/livres/out.xml");

    XMLEventReader reader = factory.createMLEventReader(new FileReader(file));
    XMLEventWriter writer = factoryOut.createMLEventWriter(new FileWriter(out));

    XMLEventFactory eventFactory = XMLEventFactory.newInstance();

    writer.add(eventFactory.createStartDocument());
    writer.add(
        eventFactory.createStartElement("livre", "http://livre.org", "newLivre"));

    while (reader.hasNext()) {
        XMLEvent next = reader.nextEvent();

        if (next.isStartElement()) {
            writer.add(next);
        } else if (next.isEndElement()) {
            writer.add(next);
        }
    }

    writer.add(eventFactory.createEndDocument());
    writer.close();
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<livre:newLivre>
  <livre xmlns="http://livre.org"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://livre.org livre.xsd">
    <titre></titre>
    <auteur>
      <nom></nom>
      <prenom></prenom>
    </auteur>
    <chapitre num="1">
      <titre></titre>
      <paragraphe>
        <titre></titre>
        <corps></corps>
      </paragraphe>
      <paragraphe>
        <titre></titre>
        <corps></corps>
      </paragraphe>
    </chapitre>
    <chapitre num="2">
      <titre></titre>
      <paragraphe>
        <titre></titre>
        <corps></corps>
      </paragraphe>
    </chapitre>
  </livre>
</livre:newLivre>
```

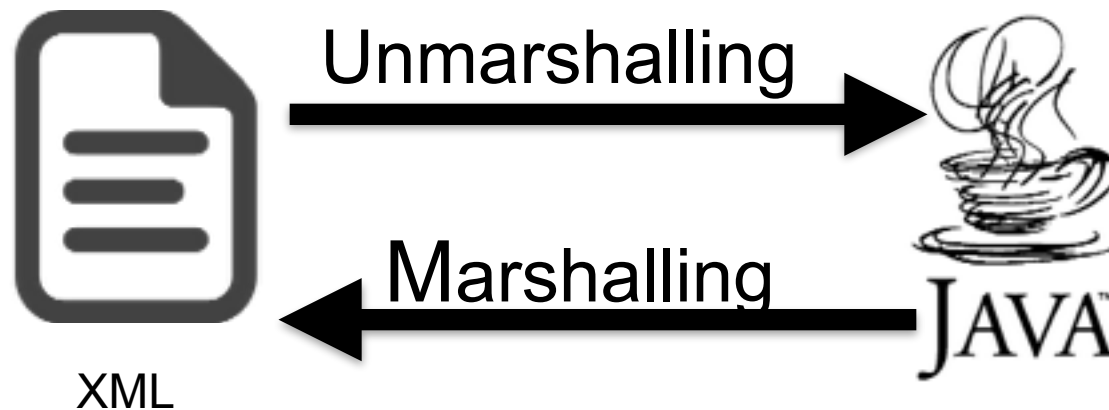

Comparatif APIs

	<i>StAX</i>	<i>SAX</i>	<i>DOM</i>
<i>API</i>	<i>Pull, streaming</i>	<i>Push, streaming</i>	<i>in memory tree</i>
<i>Simplicité</i>	<i>élevée</i>	<i>moyenne</i>	<i>élevée</i>
<i>XPath</i>	<i>non</i>	<i>non</i>	<i>oui</i>
<i>CPU / memoire</i>	<i>bon</i>	<i>très bon</i>	<i>variable</i>
<i>Retour arrière</i>	<i>non</i>	<i>non</i>	<i>oui</i>
<i>lecture XML</i>	<i>oui</i>	<i>oui</i>	<i>oui</i>
<i>écriture</i>	<i>oui</i>	<i>non</i>	<i>oui</i>
<i>update</i>	<i>non</i>	<i>non</i>	<i>oui</i>

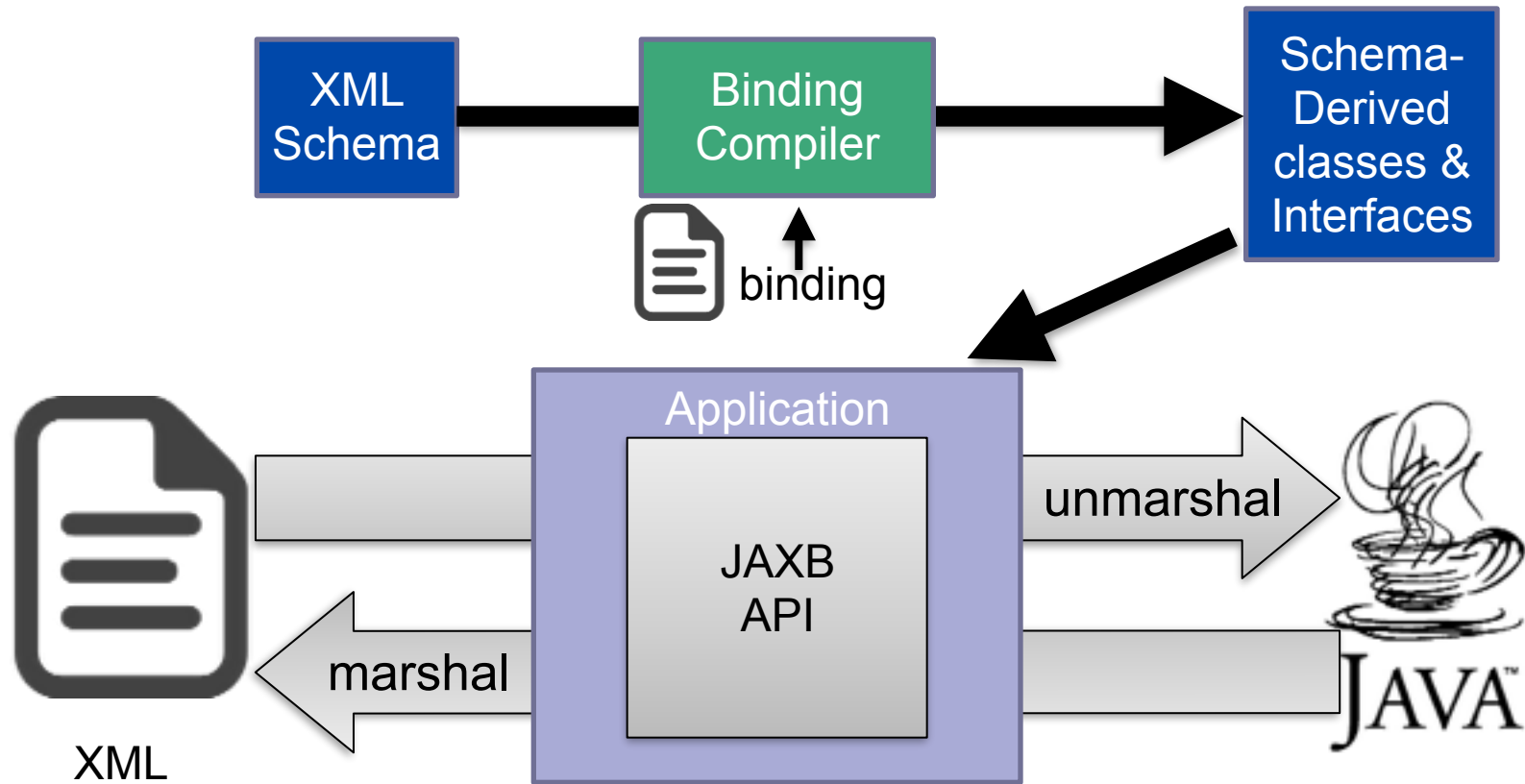
Data Binding



- Data Binding : Mapping entre un document XML et un graphe d'objets métier Java



Data Binding : Fonctionnement



Data Binding : Implémentations

- JAXB (**J**ava **A**rchitecture for **X**ML **B**inding API)
 - Apache CXF
 - Sun JAXB-RI



Data Binding : API

■ Marshalling

- JAXBContext, newInstance("package"), Marshaller, createMarshaller(), setProperty(), getNode(), getSchema(), setListener()

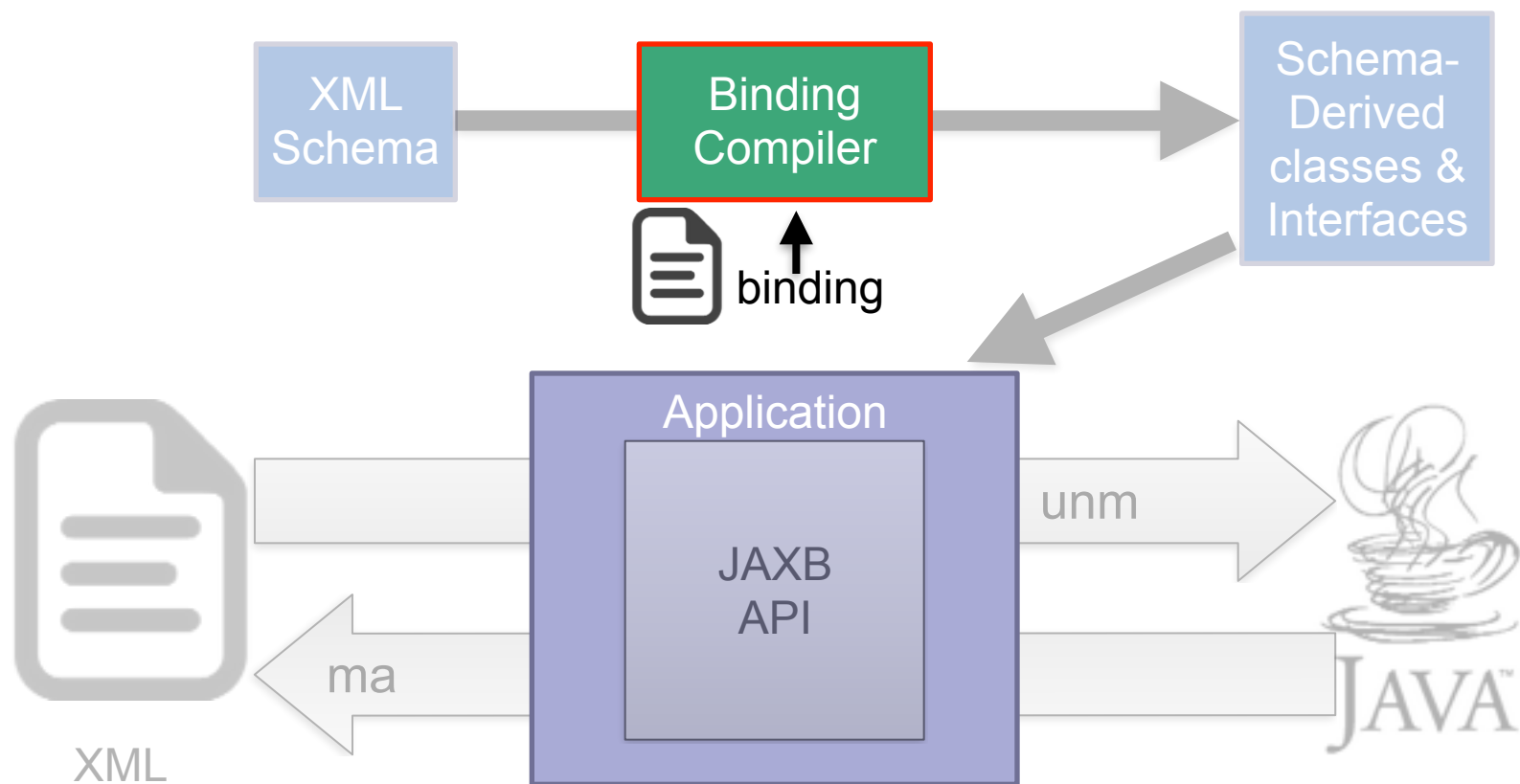
■ Unmarshalling

- JAXBContext, newInstance("package"), Unmarshaller, createUnmarshaller(), setProperty(), setSchema(), setListener()





Binding compiler



Génération de classes avec XJC

```
<?xml version="1.0"?>
<jaxb:bindings xmlns:jaxws="http://java.sun.com/xml/ns/jaxws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:jaxb="http://java.sun.com/xml/ns/jaxb"
  xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
  xsi:schemaLocation="http://java.sun.com/xml/ns/jaxb http://java.sun.com/xml/ns/jaxb/bindingschema_2_0.xsd" version="2.0">
  <jaxb:bindings schemaLocation="livre.xsd" node="/xs:schema">

    <jaxb:schemaBindings>
      <jaxb:package name="org.livre.model" />
    </jaxb:schemaBindings>

    <jaxb:globalBindings>
      <xjc:simple/>
    </jaxb:globalBindings>

    <jaxb:bindings node="xs:complexType[@name='auteurType']/xs:sequence/xs:element[@ref='livre:prenom']">
      <jaxb:property generateIsSetMethod="true" />
    </jaxb:bindings>

  </jaxb:bindings>
</jaxb:bindings>
```



Génération de classes avec XJC

- `xjc options <fichier_schéma / URL / répertoire_ou_jar>`
- Supporte
 - DTD
 - XML – Schéma
 - RELAX NG

// This file was generated by the Java™ Architecture for XML Binding(JAXB) Reference Implementation, v2.2.4-
// See http://java.sun.com/xml/jaxb
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2014-03-20 at 08:07:55 PM CET

```
package org.livre;  
import java.util.ArrayList;  
import java.util.List;  
import javax.xml.bind.annotation.XmlAccessType;  
import javax.xml.bind.annotation.XmlAccessorType;  
import javax.xml.bind.annotation.XmlElement;  
import javax.xml.bind.annotation.XmlRootElement;  
import javax.xml.bind.annotation.XmlType;  
/**  
 * <p>Java class for livreType complex type.  
 */  
@XmlAccessorType(XmlAccessType.FIELD)  
@XmlType(name = "livreType", propOrder = {  
    "titre",  
    "auteurs",  
    "chapitres"  
})  
@XmlRootElement(name = "livre")  
public class Livre {  
    @XmlElement(required = true)  
    protected String titre;  
    @XmlElement(name = "auteur", required = true)  
    protected List<Auteur> auteurs;  
    @XmlElement(name = "chapitre", required = true)  
    protected List<Chapitre> chapitres;  
    /**  
     * Gets the value of the titre property.  
     */  
    public String getTitre() {  
        return titre;  
    }  
    /**  
     * Sets the value of the titre property.  
     */  
    public void setTitre(String value) {  
        this.titre = value;  
    }  
    /**  
     * Gets the value of the auteurs property.  
     */  
    public List<Auteur> getAuteurs() {  
        if (auteurs == null) {  
            auteurs = new ArrayList<Auteur>();  
        }  
        return this.auteurs;  
    }  
    /**  
     * Gets the value of the chapitres property.  
     */  
    public List<Chapitre> getChapitres() {  
        if (chapitres == null) {  
            chapitres = new ArrayList<Chapitre>();  
        }  
        return this.chapitres;  
    }  
}
```

```
$xjc ../../src/livre.xsd -b ../../src/binding.xjb -extension  
parsing a schema...  
compiling a schema...  
org/livre/model/Auteur.java  
org/livre/model/Chapitre.java  
org/livre/model/Livre.java  
org/livre/model/ObjectFactory.java  
org/livre/model/Paragraphe.java  
org/livre/model/package-info.java
```

```
public void testJAXB() throws JAXBException {  
    File xml = new File("src/livre.xml");  
  
    JAXBContext jc = JAXBContext.newInstance("org.livre.model");  
    Unmarshaller unmarsh = jc.createUnmarshaller();  
  
    Livre livre = (Livre) unmarsh.unmarshal(xml);  
  
    System.out.println(livre.getTitre());  
    System.out.println(livre.getAuteurs().get(0).getNom()+" "+livre.getAuteurs().get(0).getPrenom());  
}
```

```
public void testJAXB() throws JAXBException {  
    File xml = new File("src/livre.xml");  
  
    JAXBContext jc = JAXBContext.newInstance("org.livre.model");  
    Unmarshaller unmarsh = jc.createUnmarshaller();  
  
    Livre livre = (Livre) unmarsh.unmarshal(xml);  
  
    System.out.println(livre.getTitre());  
    System.out.println(livre.getAuteurs().get(0).getNom()+" "+livre.getAuteurs().get(0).getPrenom());  
}
```

Apprendre le XML
Bondu Emilien

Générer le xml depuis une classe

Générer le xml depuis une classe

```
public void testJAXBWrite() throws JAXBException {
    File xml = new File("src/livreJAXB.xml");

    JAXBContext jc = JAXBContext.newInstance("org.livre.model");
    Marshaller marsh = jc.createMarshaller();
    marsh.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, Boolean.TRUE);

    Livre livre = new Livre();
    livre.setTitre("Apprendre le XML");

    Auteur auteur = new Auteur();
    auteur.setNom("Bondu");
    auteur.setPrenom("Emilien");
    livre.getAuteurs().add(auteur);

    Chapitre chap = new Chapitre();
    chap.setNum(1);
    chap.setTitre("Chapitre 1");
    Paragraphe parag = new Paragraphe();
    parag.setCorps("Contenu du premier chapitre");
    chap.getParagraphes().add(parag);

    livre.getChapitres().add(chap);

    marsh.marshal(livre, xml);

    System.out.println(livre.getTitre());
    System.out.println("Prenom ? :"+livre.getAuteurs().get(0).isSetPrenom());
}
```

Application Web...





Moteurs web

Web blog

Web mail

Forums

Wiki

Commerce électronique

Réseaux sociaux

Jeu en ligne

Messagerie instantanée

Gestion de contenus

Cloud computing:

*Le **cloud computing** est un concept qui consiste à déporter sur des serveurs distants des stockages et des traitements informatiques traditionnellement localisés sur des serveurs locaux ou sur le poste de l'utilisateur (source : wikipédia)*

Application Web *en Java*



■ Au menu

- Introduction à JEE
- Fonctionnement & structure d'une application web JEE
- Conteneur web Tomcat

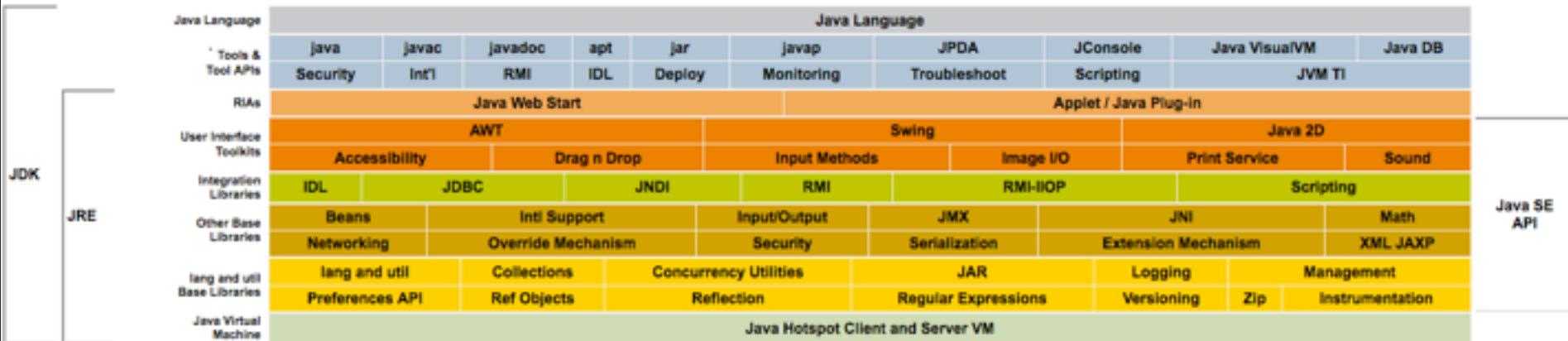




JSE vs JEE

What's the difference?

- **Java Standard Edition**

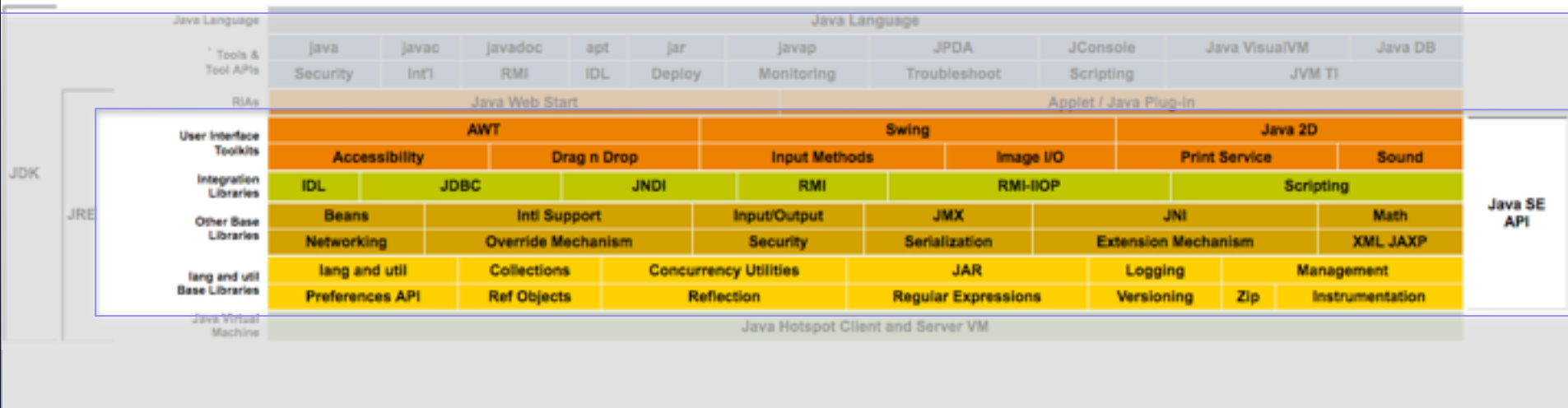




JSE vs JEE

What's the difference?

- **Java Standard Edition**



Connus des étudiants GIL ?

GUI : Swing, AWT, JAVA 2D

Librairies : JDBC

Base : Math, I/O, **XML-JAXP**, Serialization, Beans



JSE vs JEE

What's the difference?

- **Java Standard Edition**

... pour les applications desktop

... et Java Web Start ?

lance une application java depuis un navigateur

... et Java Applet ?

source : <http://java.sun.com/applets/jdk/1.4/demo/applets/>

A Clock (1.6)



java, mars 22 10:10:10 2012



JSE vs JEE

What's the difference?

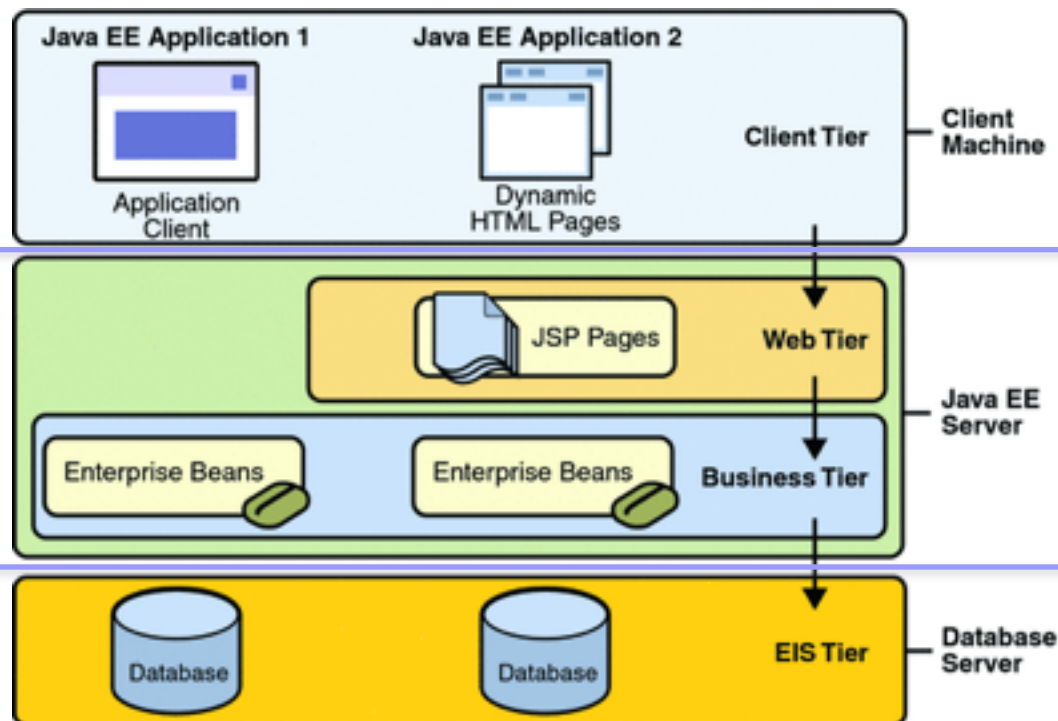
■ Java Enterprise Edition

... pour les applications multi-tiers

1

2,3

4





JSE vs JEE

What's the difference?



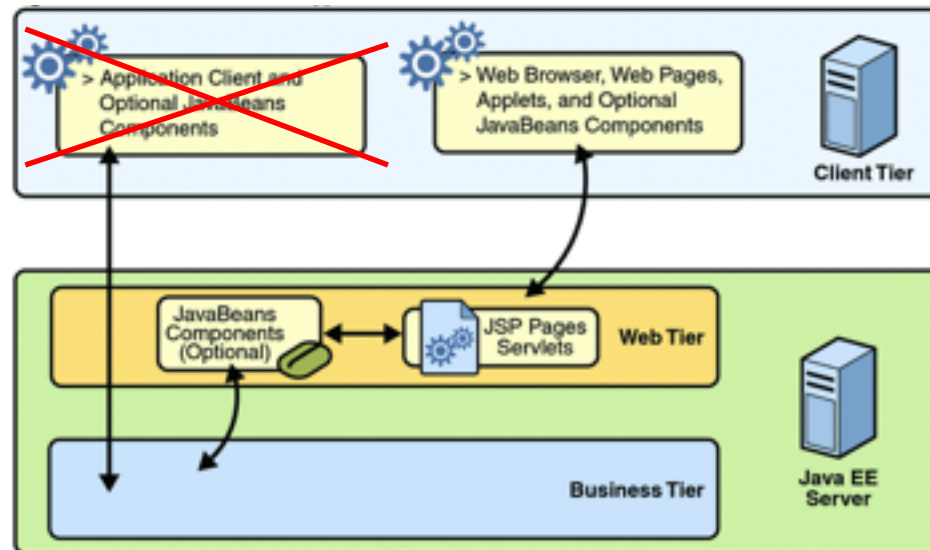
Java Enterprise Edition

... pour les applications web

■ Java EE

... pour les GIL

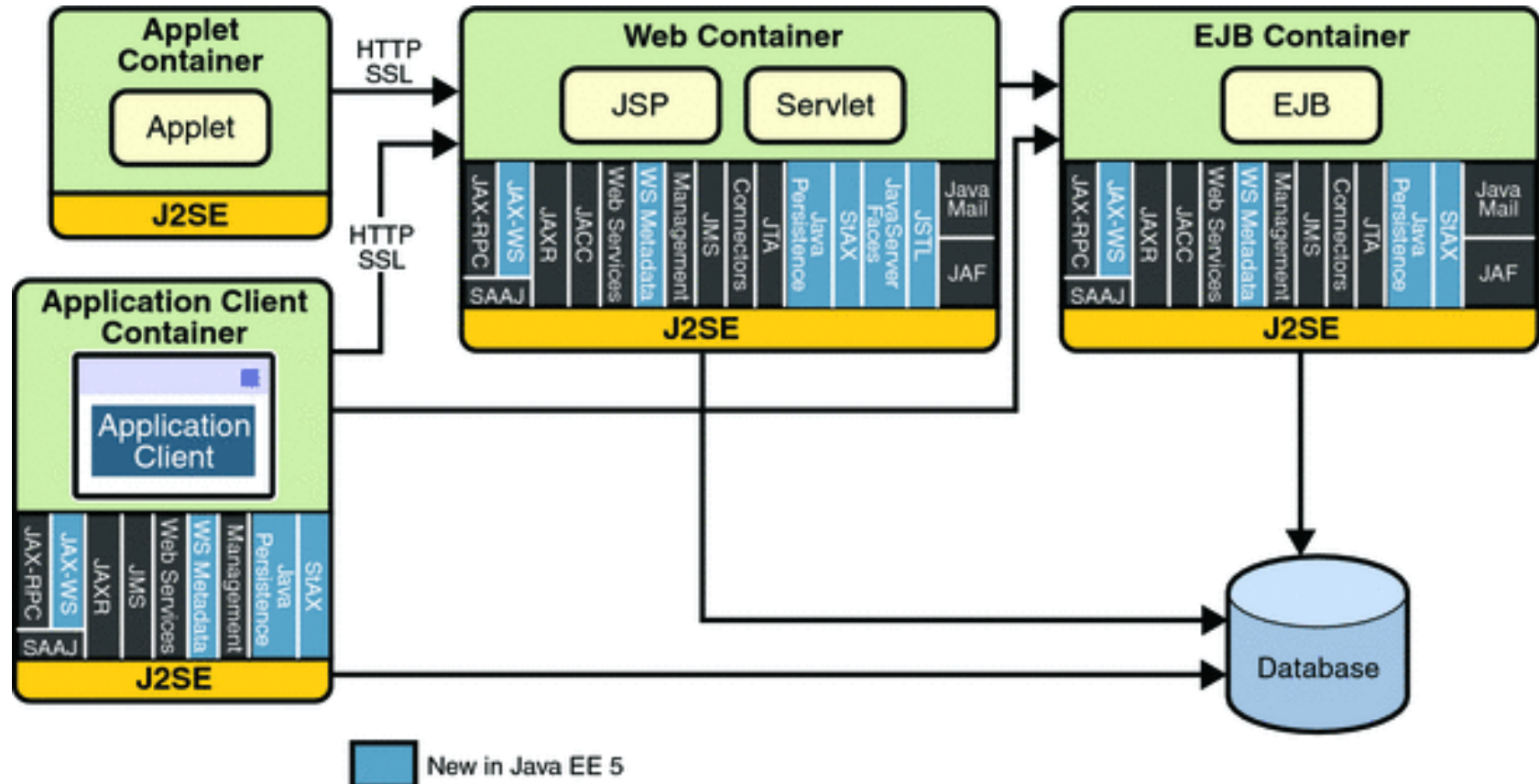
- Le **client** = navigateur Web
- Le **Web Tier** = Servlet + JSP + Beans
- Le **métier** = gérer des CV
- Le **serveur JEE** = Tomcat



Java EE 5 *(2006)*





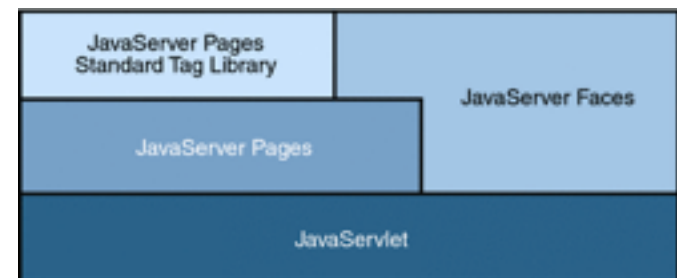
Vue d'ensemble :



Java EE 5

... pour les GIL

- Conteneur Web
- JavaServer Page (JSP)
- Servlet API - [javax.servlet](http://java.sun.com/javax.servlet)
- Standard Tag Library (JSTL)
- JavaServer Faces (JSF)
- Java API for XML Processing (JAXP) 
- Java Architecture for XML Binding (JAXB) 



Web application

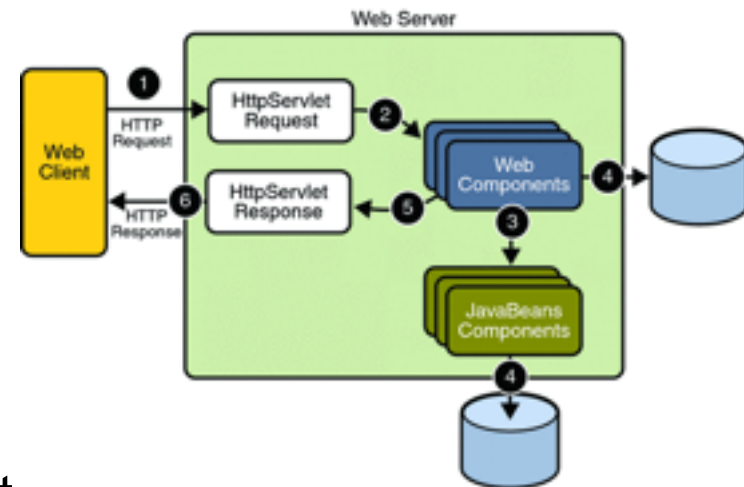
Fonctionnement de base

Traitement des requêtes :

- 1) Conversion des requêtes HTTP clientes
- 2) Envoi des requêtes aux composants
- 3) Interactions avec des beans
- 4) Interactions avec les données

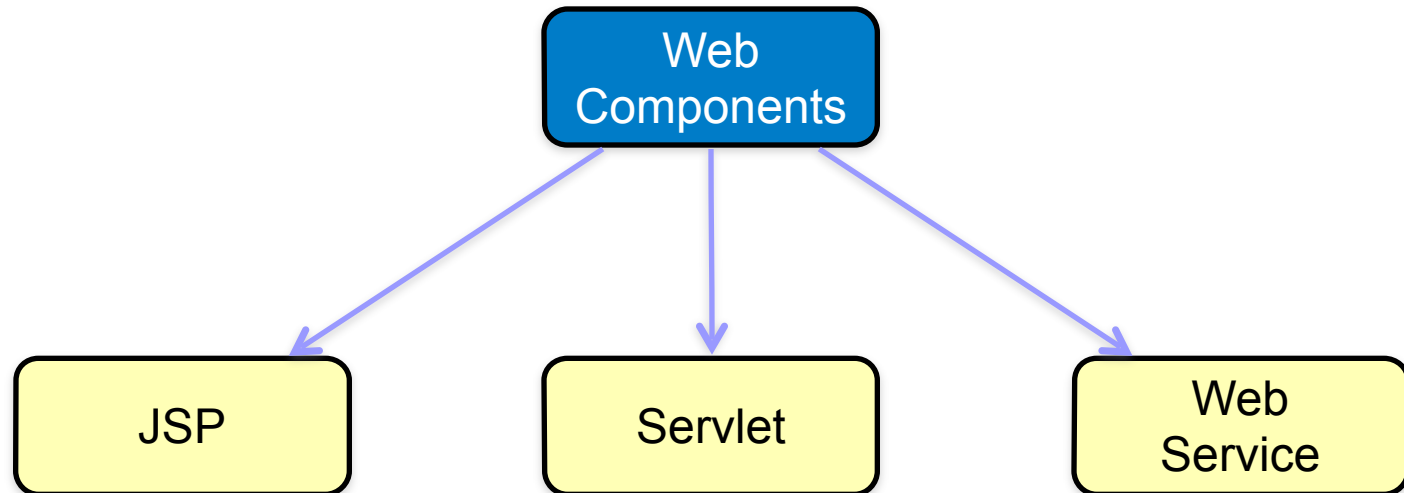
Distribution de la requête aux autres composants

- 1) Construction de la réponse
- 2) Conversion et envoi de la réponse au client



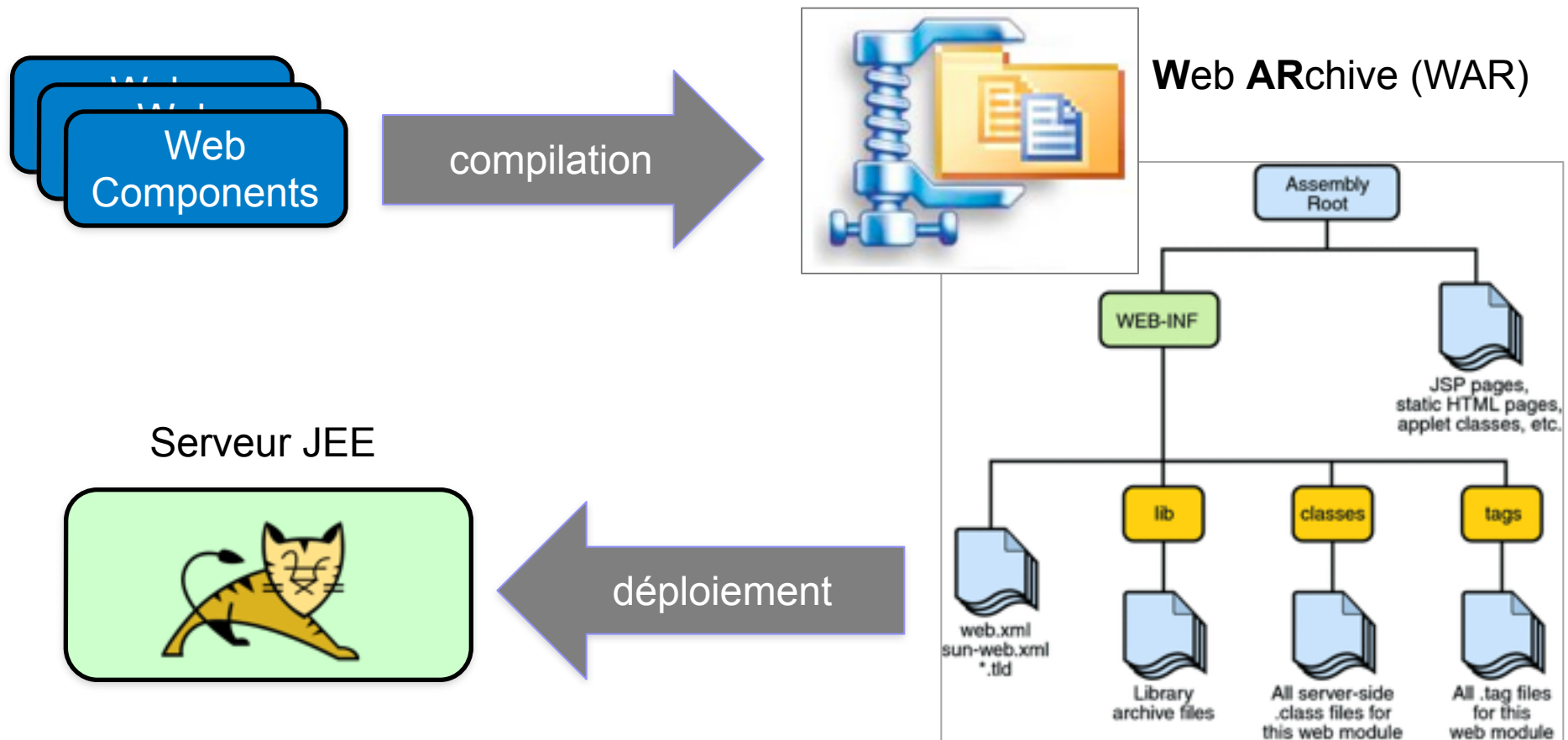
Application web = composant **S** web

Web components



Application web = composant **S** web

Web components



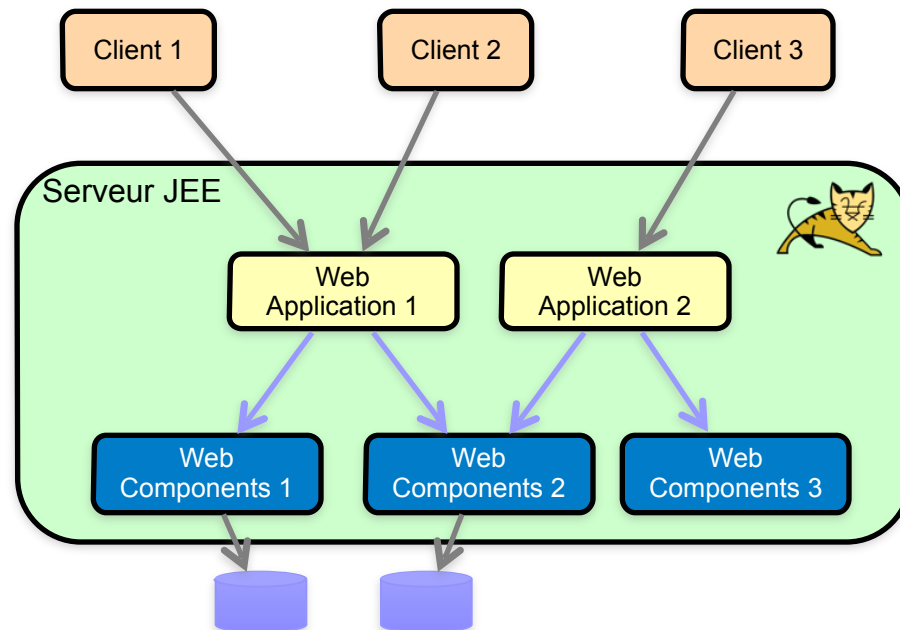
Application web = composant **S** web

Architecture JEE

Conclusion

Un serveur = **des** applications**S**

Une application = **des** composants**S**



Serveur d'application web Tomcat



- Serveur d'applications web d'apache
 - (servlet / jsp / services web)
- Implémente **une partie** des spécifications JEE
- EJB non implémenté

- Conteneurs certifiés JEE :



Tomcat : server



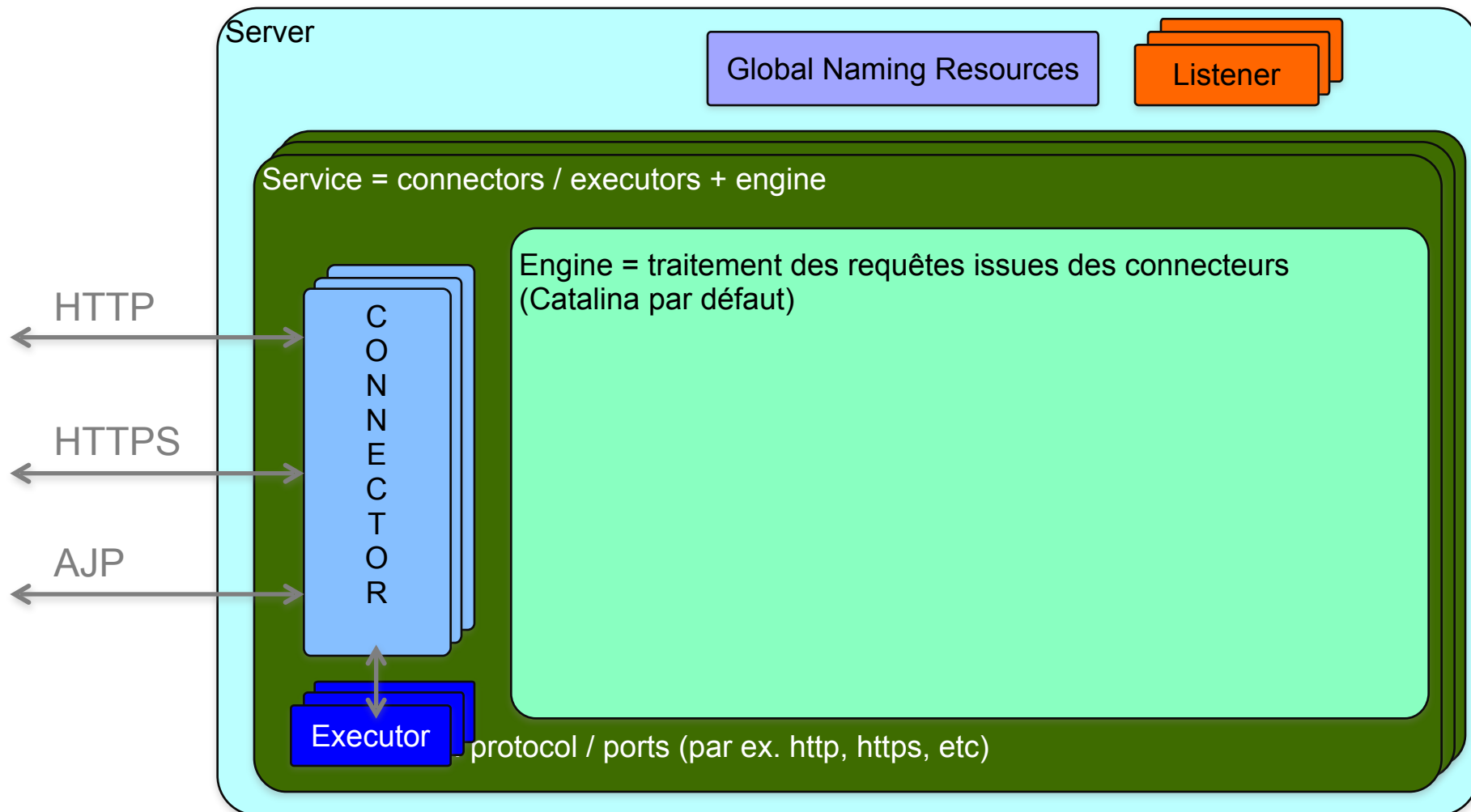
Server ≠ container

Global Naming Resources

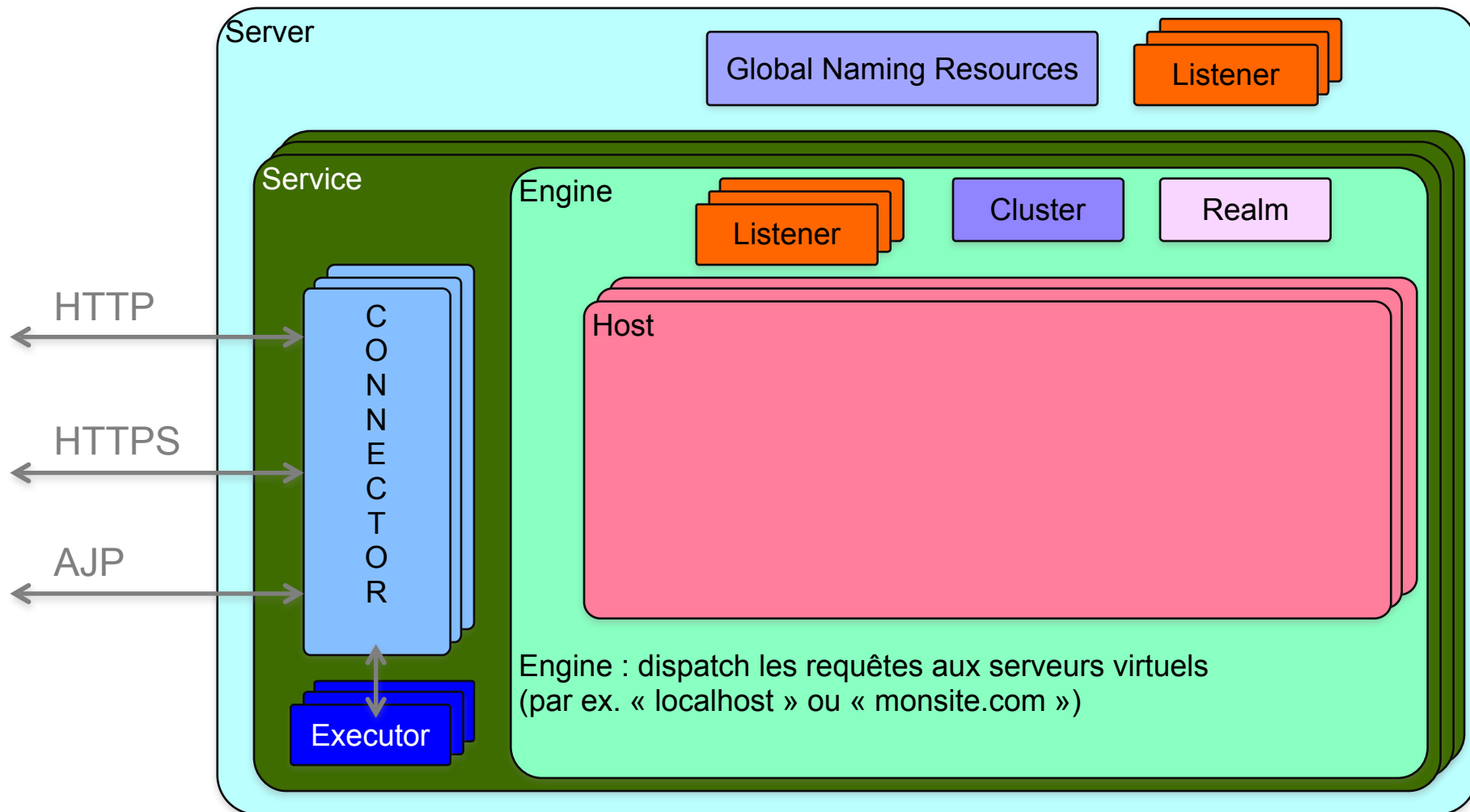
Listener

- Port : port TCP/IP l'extinction du serveur (8005)
- Shutdown : commande pour l'extinction (SHUTDOWN)
config : server.conf
- Java Naming Directory Interface (JNDI) : déclaration des ressources du serveur (par ex. la DB des utilisateurs du serveur)
- Listener : classes qui écoutent le serveur (par ex. pour la libération de la mémoire)

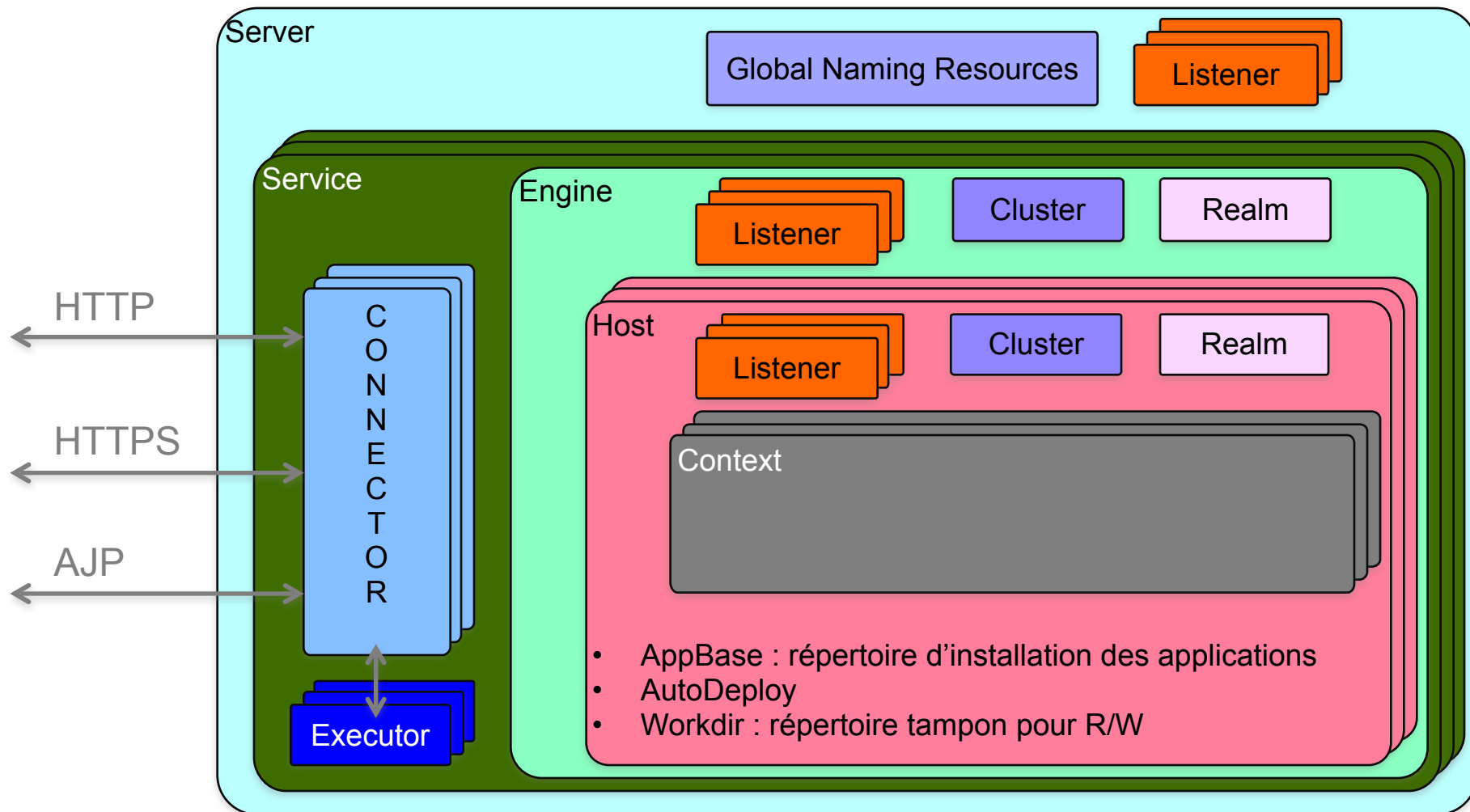
Tomcat : service



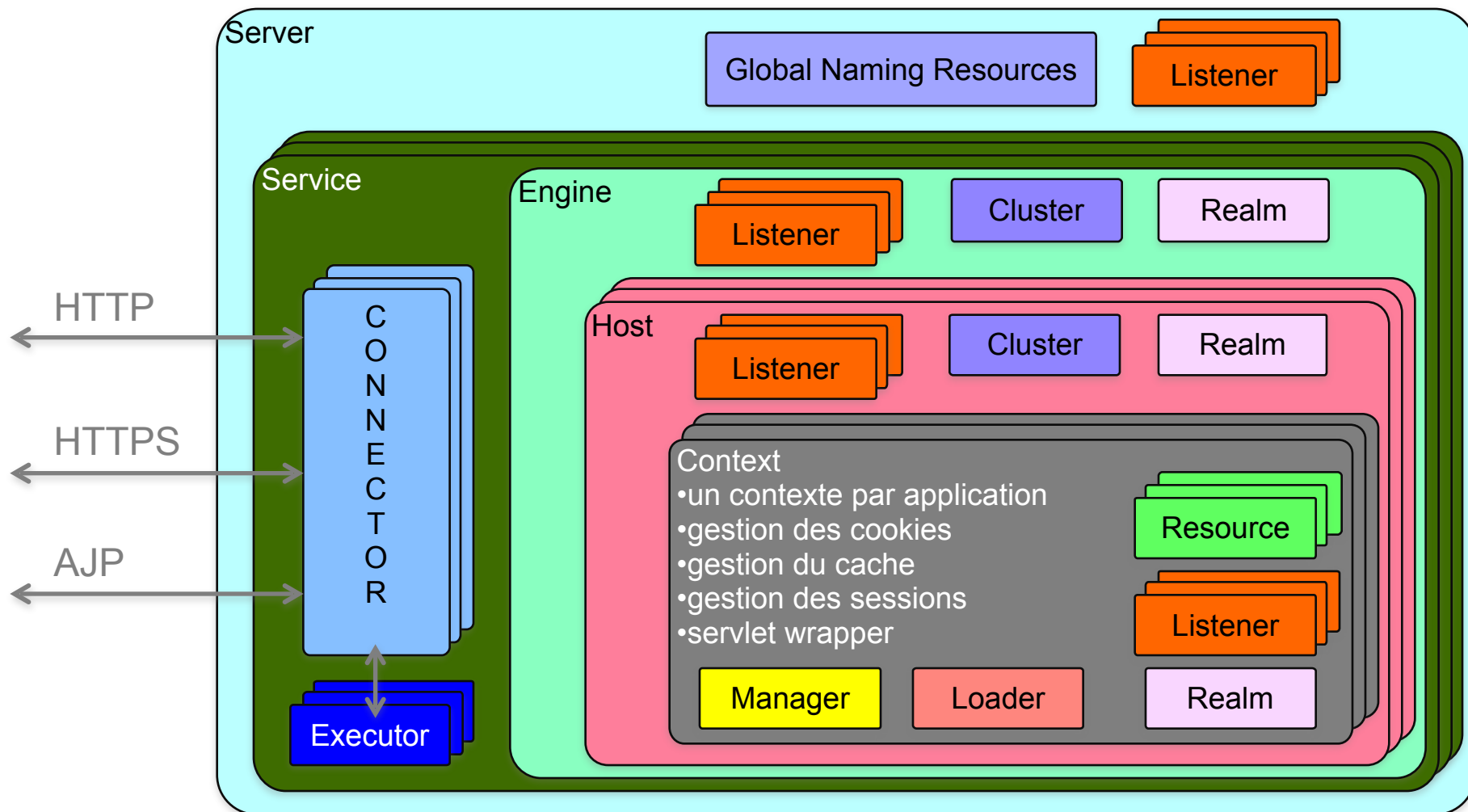
Tomcat : engine



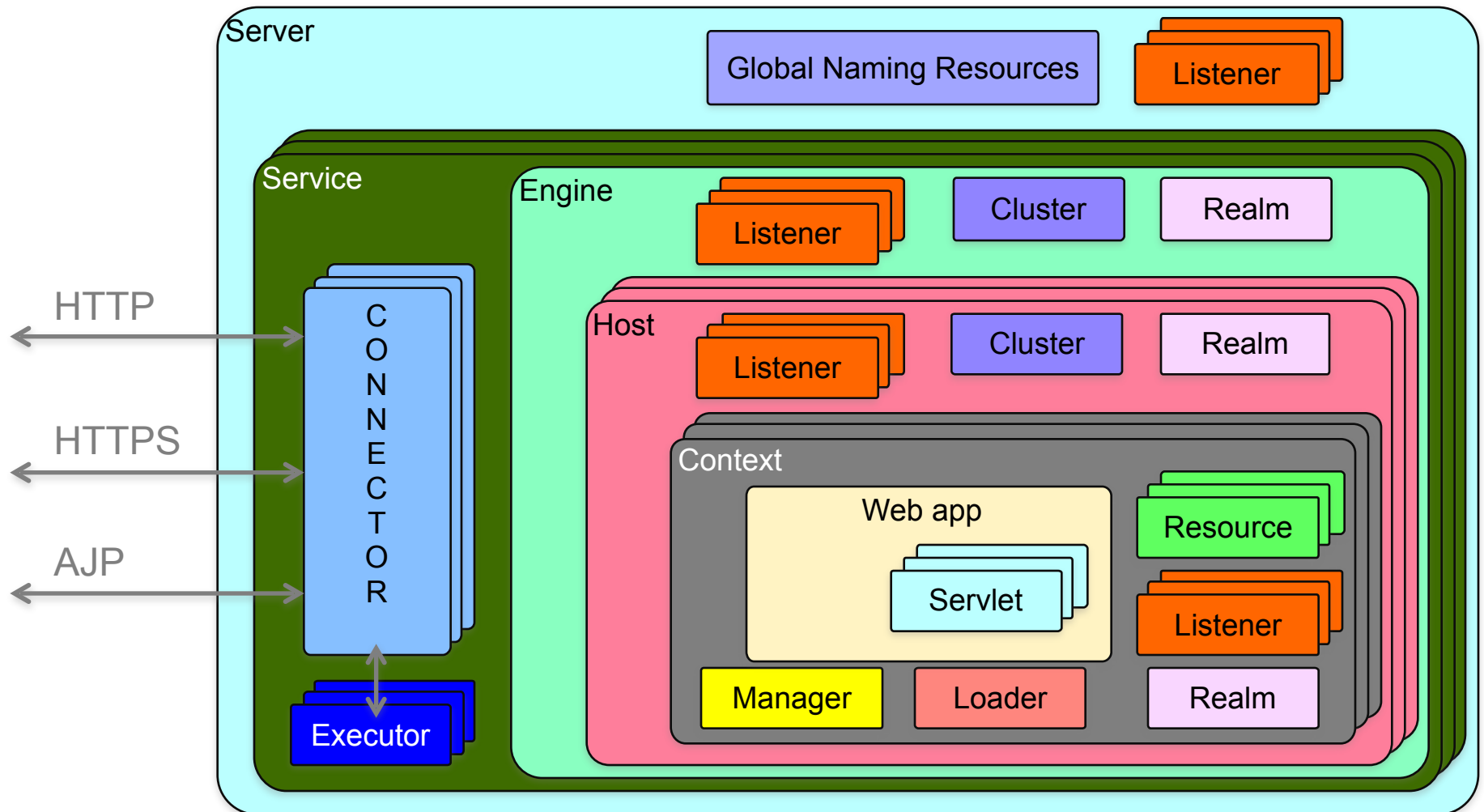
Tomcat : host



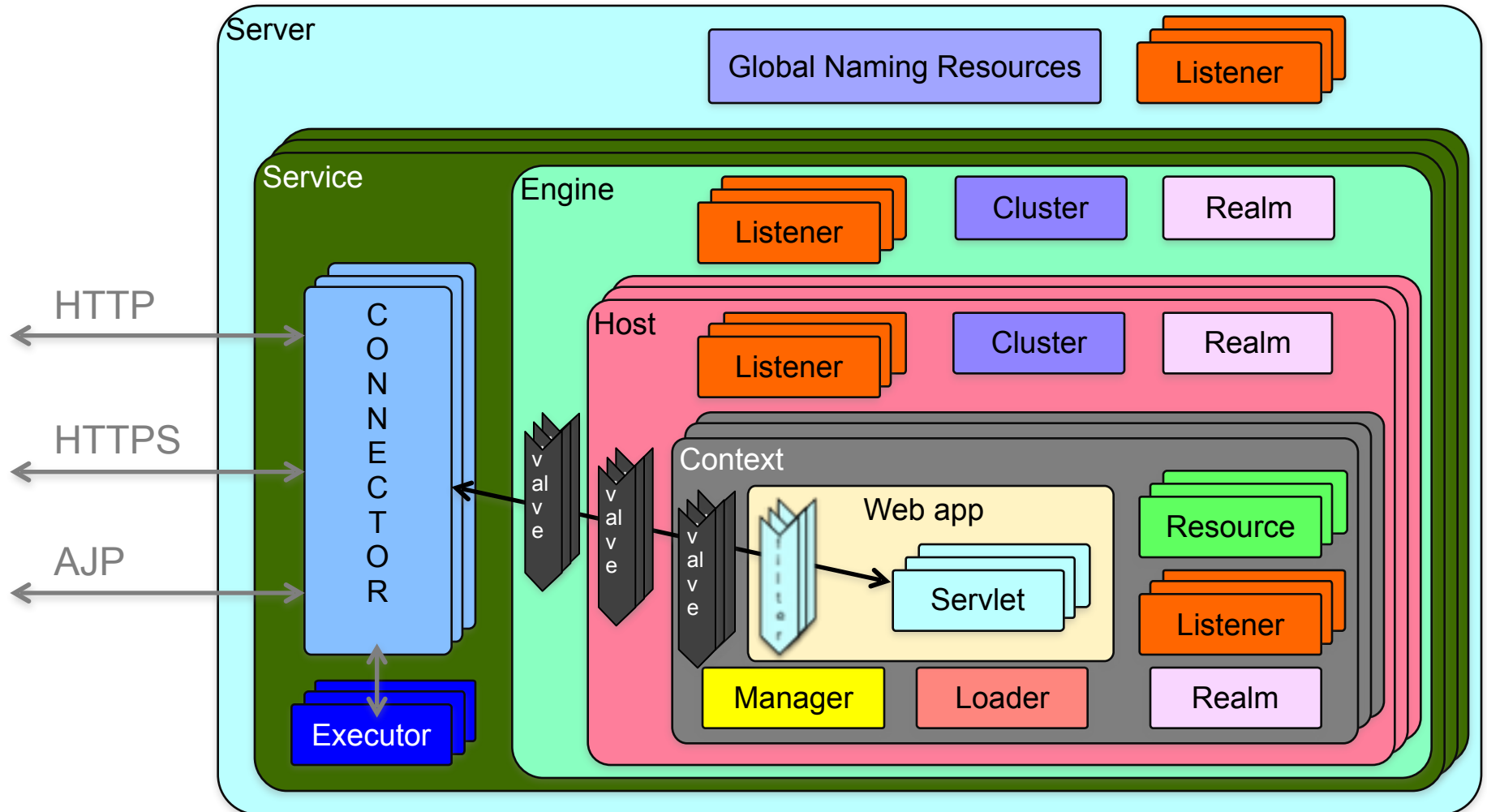
Tomcat : context



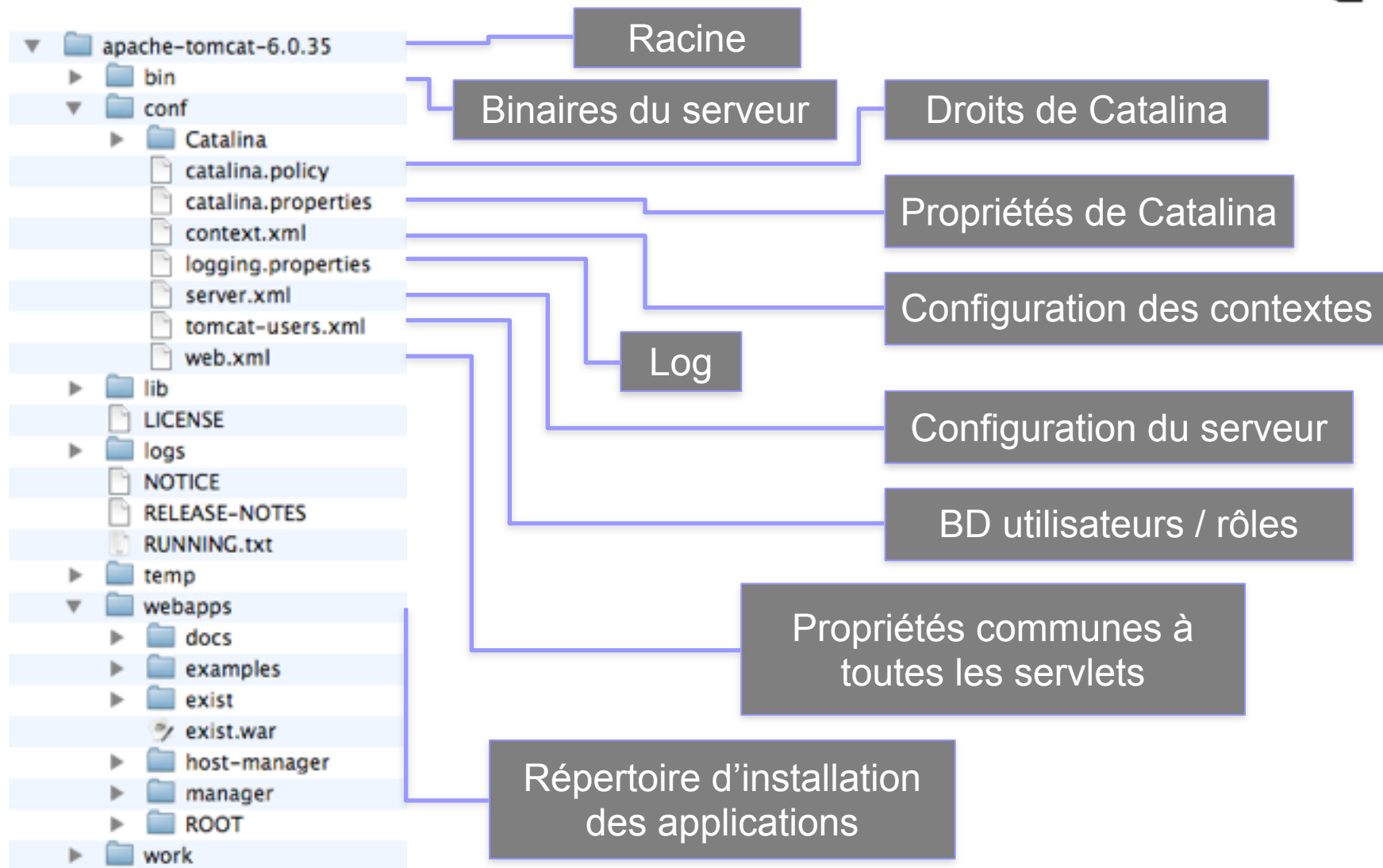
Tomcat : Web app



Tomcat : propagation des requêtes – « robinets »



Tomcat : arborescence



Tomcat : configuration



```
<?xml version="1.0" encoding="UTF-8"?>
<!-- SERVER --><Server port="8005" shutdown="SHUTDOWN">

  <!-- SERVER LISTENER -->
  <Listener SSLEngine="on" className="org.apache.catalina.core.AprLifecycleListener"/>
  <Listener className="org.apache.catalina.core.JasperListener"/>
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener"/>
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener"/>
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener"/>

  <!-- GLOBAL RESOURCES -->
  <GlobalNamingResources>
    <!-- USER DB -->
    <Resource auth="Container" description="User database that can be updated and saved" factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
name="UserDatabase" pathname="conf/tomcat-users.xml" type="org.apache.catalina.UserDatabase"/>
  </GlobalNamingResources>

  <!-- SERVICE PROVIDED BY THE SERVER-->
  <Service name="Catalina">

    <!-- ASSOCIATED CONNECTORS -->
    <Connector connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>
    <Connector port="8009" protocol="AJP/1.3" redirectPort="8443"/>

    <!-- ASSOCIATED ENGINE -->
    <Engine defaultHost="localhost" name="Catalina">

      <!-- ROLES & USER MANAGEMENT -->
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm" resourceName="UserDatabase"/>

      <!-- DEFAULT HOST FOR THIS ENGINE -->
      <Host appBase="webapps" autoDeploy="true" name="localhost" unpackWARs="true" xmlnsNamespaceAware="false" xmlValidation="false">

        <!-- LOGGER VALVE -->
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs" pattern="common" prefix="localhost_access_log." resolveHosts="false"
suffix=".txt"/>
      </Host>
    </Engine>
  </Service>
</Server>
```

Tomcat : déploiement

des applications



Choix 1 - upload

Deployer

Emplacement du répertoire ou fichier WAR de déploiement sur le serveur

Chemin de contexte (requis):

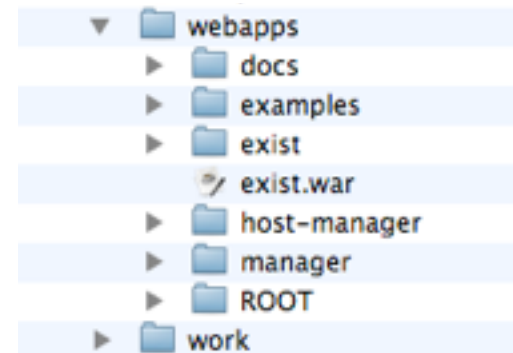
URL du fichier XML de configuration:

URL vers WAR ou répertoire:

Fichier WAR à déployer

Choisir le fichier WAR à téléverser

Choix 2 – copier/coller

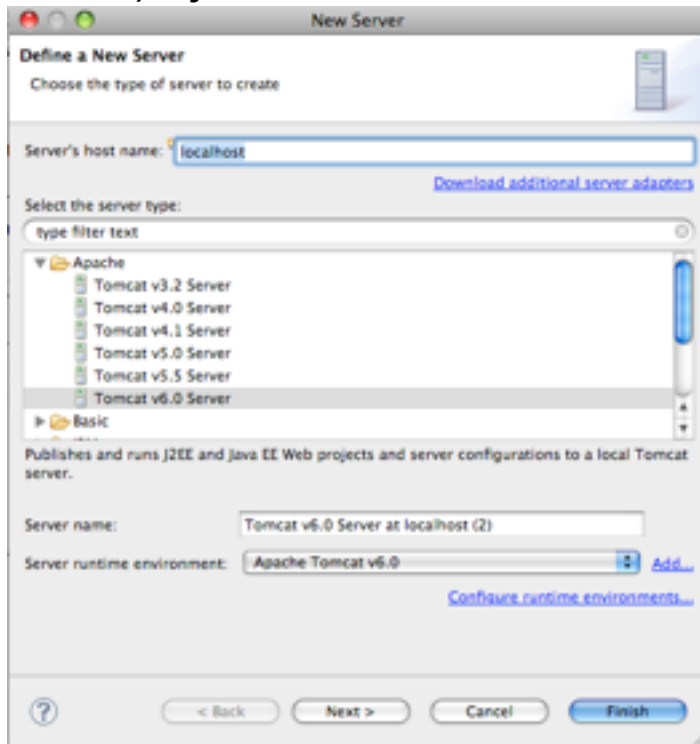


Tomcat

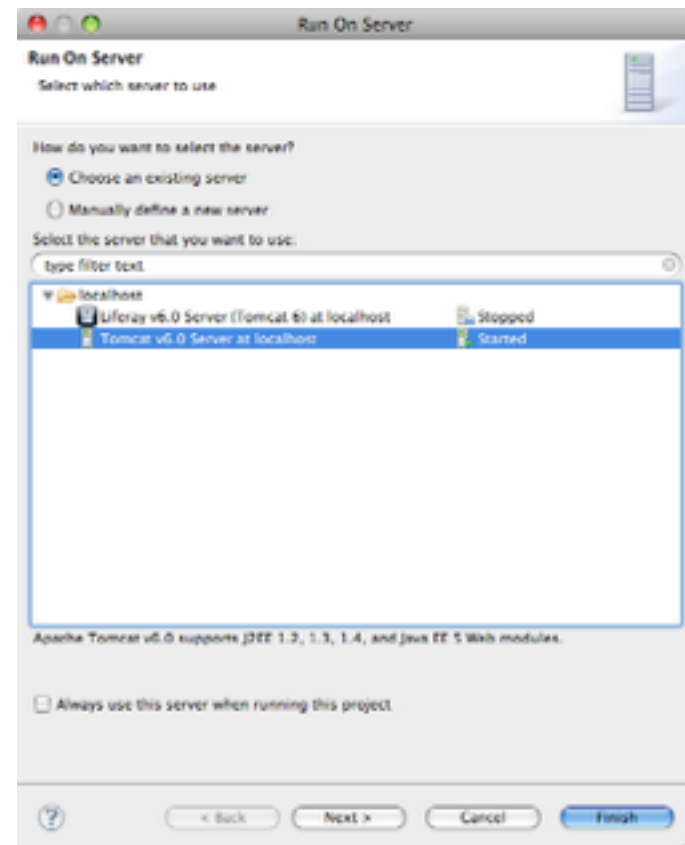
dans un IDE



1) Ajouter d'un serveur



2) Exécuter une application sur le serveur



Tomcat

dans un IDE (vue Java EE)



Java EE - http://localhost:8080/CVTheque/ - Eclipse - /Users/emilienbondu/Developpement/workspace/Master

Project Explorer

- cv_web_service
- cv_xml_tp4
- CVTheque
 - Deployment Descriptor: CVTheque
 - Java Resources
 - JavaScript Resources
 - build
 - main
 - src
 - main
 - java
 - univ
 - rouen
 - fr
 - CVThequeSearch.java
 - CVThequeUpload.java
 - resources
 - cv_frag.xml
 - cv.xml
 - cvtheque_en.properties
 - cvtheque_fr.properties
 - query_age.xql
 - query_filter_competence.xql
 - WebContent
 - css
 - jsp
 - search.jsp
 - upload.jsp
 - META-INF
 - WEB-INF

cv.xml CVThequeSearch.java CVTheque - Search

http://localhost:8080/CVTheque/search

Rechercher des cv
Age sup à : Rechercher
[Ajouter des CVs dans la base](#)

Outline Task List

An outline is not available.

Markers Properties Servers Snippets Console

Tomcat v6.0 Server at localhost [Apache Tomcat] /System/Library/Frameworks/JavaVM.framework/Versions/1.6.0/Home/bin/java (27 mars 2012)

```
27 mars 2012 12:01:07 org.apache.coyote.http11.Http11Protocol init
INFO: Initialisation de Coyote HTTP/1.1 sur http-8080
27 mars 2012 12:01:07 org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 912 ms
27 mars 2012 12:01:07 org.apache.catalina.core.StandardService start
INFO: Démarrage du service Catalina
27 mars 2012 12:01:07 org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.32
27 mars 2012 12:01:07 org.apache.coyote.http11.Http11Protocol start
INFO: Démarrage de Coyote HTTP/1.1 sur http-8080
27 mars 2012 12:01:07 org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
27 mars 2012 12:01:07 org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/42 config=null
27 mars 2012 12:01:07 org.apache.catalina.startup.Catalina start
INFO: Server startup in 762 ms
```

Tomcat

dans un IDE (vue debug)



Debug - CVTheque/src/main/java/univ/rouen/fr/CVThequeSearch.java - Eclipse - /Users/emilienbondu/Developpement/workspace/Master

Debug

Daemon Thread [http-8080-1] (Suspended (breakpoint at line 81 in CVThequeSearch.java))

CVThequeSearch.doPost(HttpServletRequest, HttpServletResponse) line: 81

CVThequeSearch(HttpServletRequest).service(HttpServletRequest, HttpServletResponse) line: 717

CVThequeSearch(HttpServletRequest).service(ServletRequest, ServletResponse) line: 717

ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 290

ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 206

StandardWrapperValve.invoke(Request, Response) line: 233

StandardContextValve.invoke(Request, Response) line: 191

StandardHostValve.invoke(Request, Response) line: 127

Variables

Name	Value
this	CVThequeSearch (id=45)
request	RequestFacade (id=60)
response	ResponseFacade (id=64)
age	"25" (id=68)
col	null

25

Outline

- univ.rouen.fr
 - import declarations
 - CVThequeSearch
 - serialVersionUID : long
 - URI : String
 - driver : String
 - collectionName : String
 - dbadmin_user : String
 - init() : void
 - doGet(HttpServletRequest, HttpServletResponse) : void
 - doPost(HttpServletRequest, HttpServletResponse) : void
 - createCollection() : void

Console

Tomcat v6.0 Server at localhost [Apache Tomcat] /System/Library/Frameworks/JavaVM.framework/Versions/1.6.0/Home/bin/java (27 mars 2012 12:08:01)

27 mars 2012 12:08:03 org.apache.catalina.startup.Catalina start

INFO: Server startup in 787 ms

univ.rouen.fr.CVThequeSearch.doPost(HttpServletRequest, HttpServletResponse) line: 81