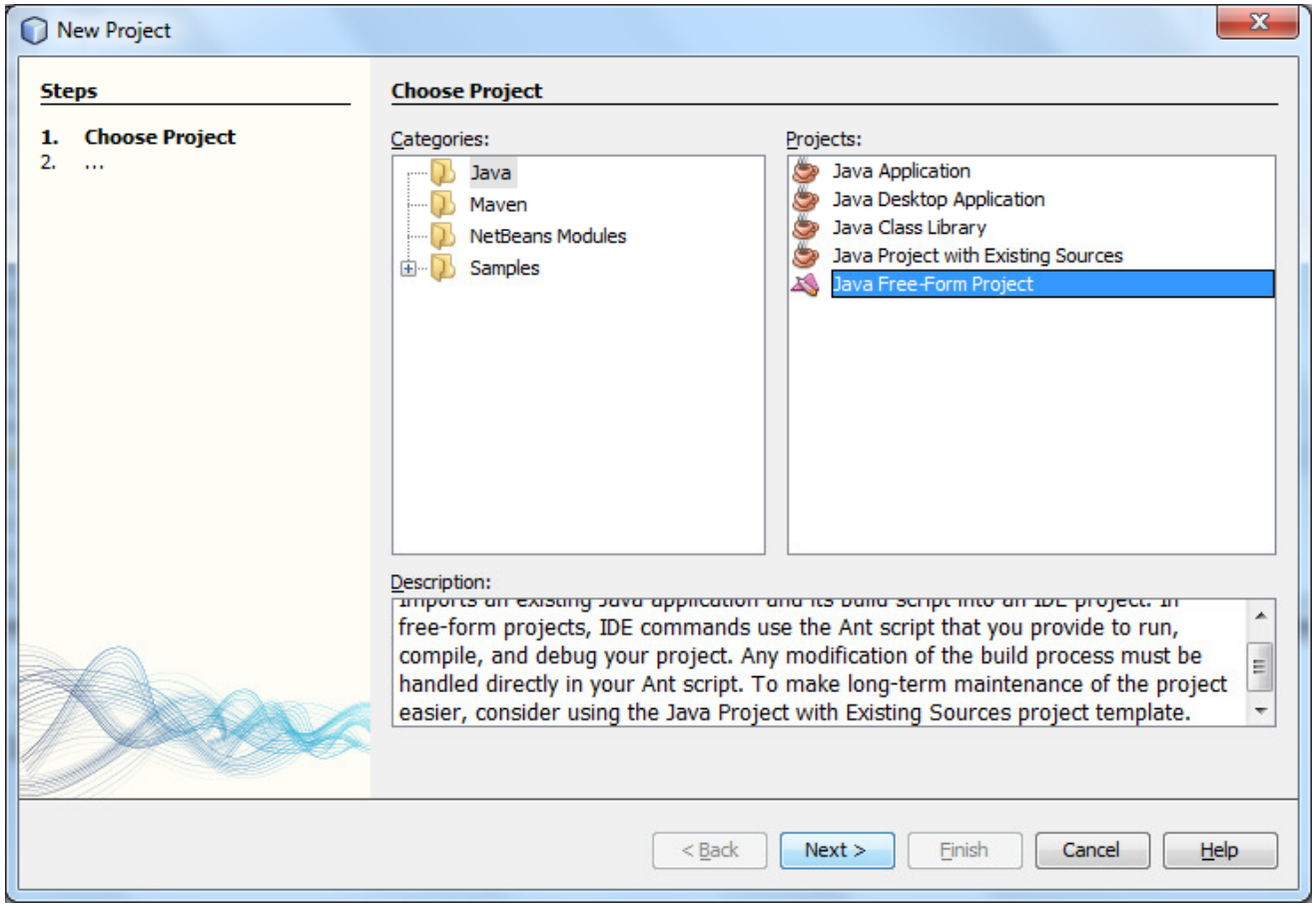
Medii de programare în Inteligența Artificială

Embedding JESS with JAVA

Lect. univ. dr. Mihaela Colhon

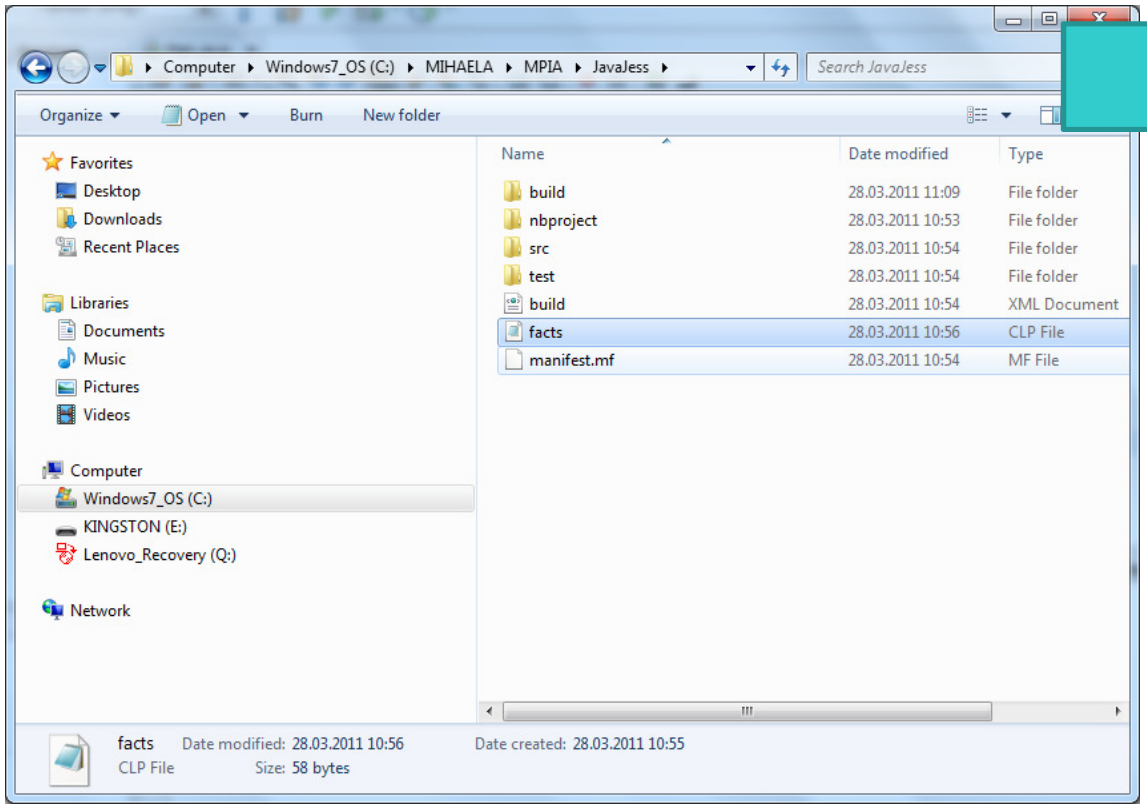
http://inf.ucv.ro/~ghindeanu

Pasul 1. Cream un proiect nou Java



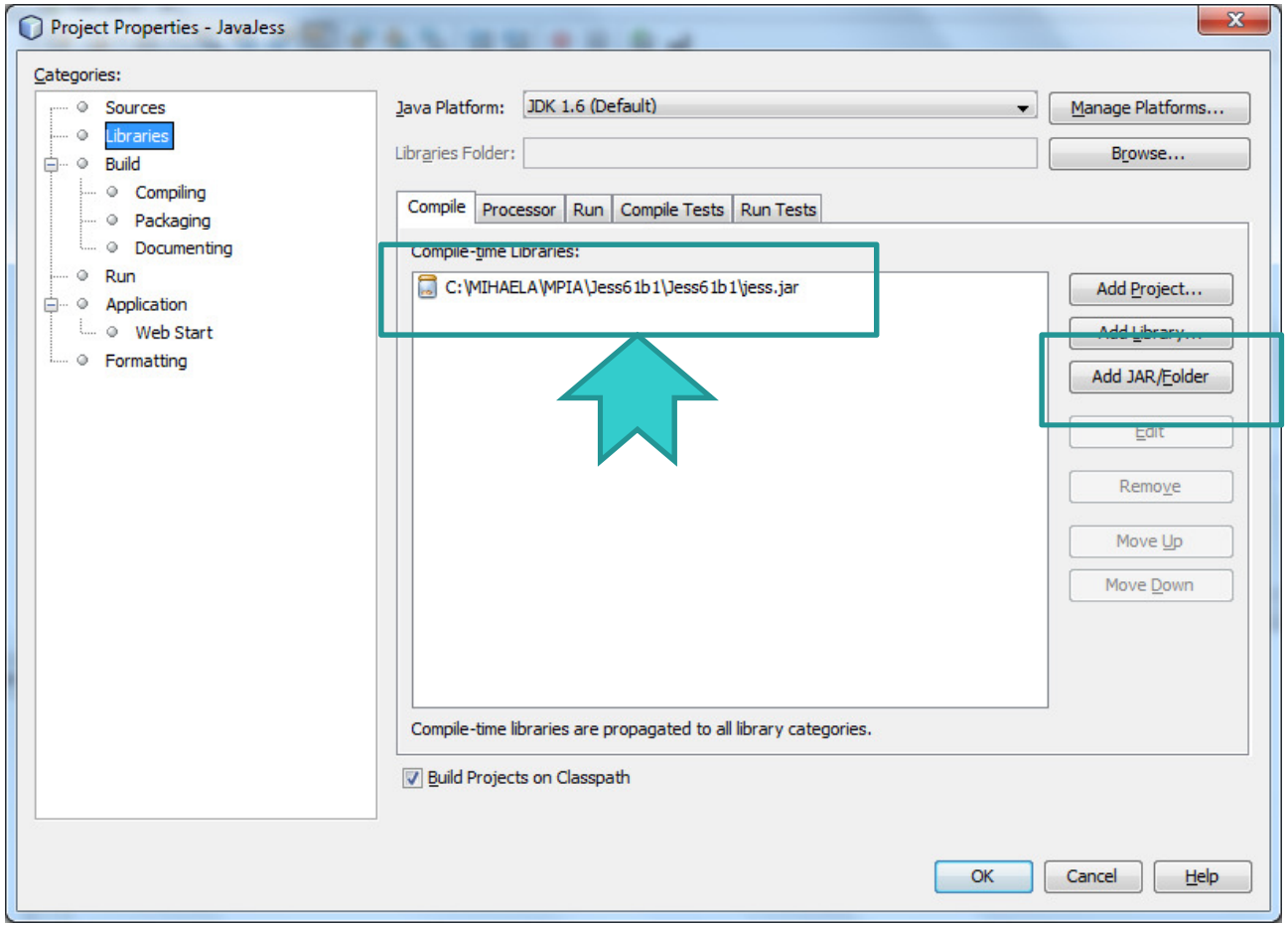
Pasul 2. In directorul proiectului cream un fisier .clp

(printout t ″Hello from Jess in Java application″ crlf)



facts.clp

Pasul 3. Includem libraria Jess in proiectul Java



Pasul 4. In clasa principala a proiectlui incarcam fisierul .clp

Main.java

```
6   package javajess;
7
9   import jess.Deffacts;
    import jess.JessException;
    import jess.Fact;
    import jess.RU;
12  import jess.Rete;
    import jess.Value;
14
15
16  public class Main {
17
18      
19      public static void main(String[] args) throws JessException{
20          //String pentru stocarea comenzilor Jess
21          String buff;
22          //cream o instanta a clasei de inferfata cu Jess
23          Rete engine = new Rete();
24          engine.executeCommand("(batch facts.clp)");
25          engine.reset();
26
27
28      }
29
30  }
31
```

Pasul 5. Rulam aplicatia Java

Test Results | Output - JavaJess (run)

```
run:
Hello from Jess in Java application!
BUILD SUCCESSFUL (total time: 0 seconds)
```

Rulare cod Jess din aplicatii Java

Jess vine cu un pachet de clase Java (\jess_distribution\Jess61p4\docs\api.html) prin intermediul carora se pot accesa functii Jess pentru o aplicatie Java. Aceste clase se gasesc in arhiva jess.jar.

Cel mai simplu mod de a executa cod Jess din Java implica urmatorii pasi:

1. Se scrie codul Jess intr-un fisier .clp
2. Se creaza in aplicatia Java o instanta a clasei Rete.
3. Se invoca functia batch prin intermediul functiei executeCommand a clasei Rete.

```
import jess.Rete;
public class Demo {

 public static void main(String[] args) {
  (new Demo()).callJessFile();
 }

 public void callJessFile() {
   Rete engine = new Rete();
   try {
    engine.executeCommand("(batch facts.clp)");
    engine.reset();
   }
   catch(Exception e)
                {e.printStackTrace(); }
 }
 }
```

Incarcarea de cunostinte noi in Java. Exemplificare

Sa consideram urmatorul cod Jess:

facts - Notepad

File Edit Format View Help

```
(printout t "Hello from Jess in Java application!" crlf)

(deftemplate personal-data
        (slot name)
        (slot age)
        (slot weight)
        (slot height)
        (slot processed (default FALSE))
)

(deffacts people
        (personal-data (name Andrei) (age 20) (weight 80) (height 188))
        (personal-data (name Corina) (age 22) (weight 60) (height 168))
)

(defrule birthday
        ?birthday <- (birthday ?name)
        ?data-fact <- (personal-data (name ?name) (age ?age) (weight 80) (height 188))
=>
(modify ?data-fact (processed TRUE))
(retract ?birthday)
)
```

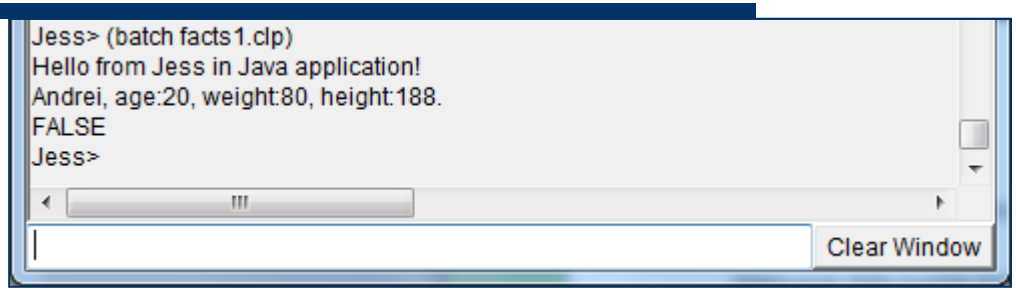

public class Main {
public static void main(String[] args) throws JessException{
//String pentru stocarea comenzilor Jess
String buff;
//cream o instanta a clasei de inferfata cu Jess
Rete engine = new Rete();
engine.executeCommand("(batch facts.clp)");
engine.reset();
engine.executeCommand("(facts)");
engine.executeCommand("(watch facts)");
engine.executeCommand("(assert (birthday Andrei))");
engine.run();
engine.executeCommand("(facts)");
st Results
Output - JavaJess (run)
run:
Hello from Jess in Java application!
f-0 (MAIN::initial-fact)
f-1 (MAIN::personal-data (name Andrei) (age 20) (weight 80) (height 188) (processed FALSE))
f-2 (MAIN::personal-data (name Corina) (age 22) (weight 60) (height 168) (processed FALSE))
For a total of 3 facts.
==> f-3 (MAIN::birthday Andrei)
<=> f-1 (MAIN::personal-data (name Andrei) (age 20) (weight 80) (height 188) (processed TRUE))
<== f-3 (MAIN::birthday Andrei)
f-0 (MAIN::initial-fact)
f-1 (MAIN::personal-data (name Andrei) (age 20) (weight 80) (height 188) (processed TRUE))
f-2 (MAIN::personal-data (name Corina) (age 22) (weight 60) (height 168) (processed FALSE)
For a total of 3 facts.
BUILD SUCCESSFUL (total time: 0 seconds)

Comanda Defquery

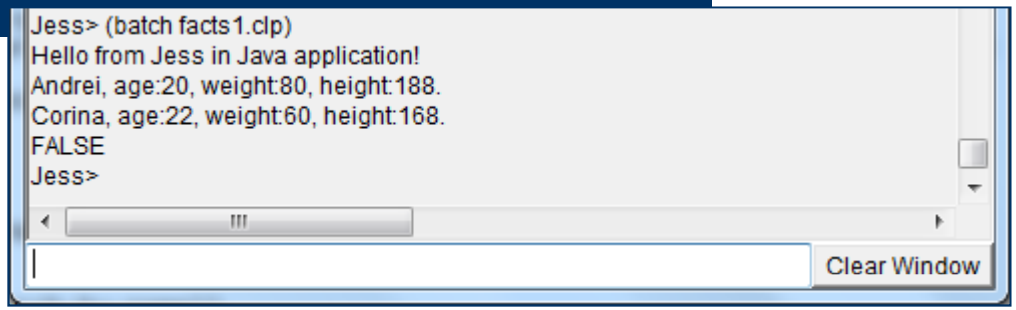
Constructia defquery permite construirea unei reguli care are nu are membru drept. Spre deosebire de reguli care se activeaza automat in functie de continutul memoriei, interogarile definite cu defquery sunt in totalitate sub controlul programatorului. O astfel de constructie returneaza intr-un obiect de tipul java.util.Iterator toate potrivirile care respecta conditiile din interogare.

Comanda defquery/run-query. Exemplu 1

```
(deftemplate personal-data
                (slot name)
                (slot age)
                (slot weight)
                (slot height)
                (slot processed (default FALSE)))
(deffacts people
                (personal-data (name Andrei) (age 20) (weight 80) (height 188))
                (personal-data (name Corina) (age 22) (weight 60) (height 168)))
(defquery search-by-name
                (declare (variables ?n))
                (personal-data (name ?n) (age ?a) (weight ?w) (height ?h)))
(reset)
(bind ?res (run-query search-by-name Andrei))
(while (?res hasNext)
                (bind ?fact (call (call ?res next) fact 1))
                (bind ?n (call ?fact get 0))
                (bind ?a (call ?fact get 1))
                (bind ?w (call ?fact get 2))
                (bind ?h (call ?fact get 3))
                (printout t ?n ", age:" ?a ", weight:" ?w ", height:" ?h "." crlf))
```



Comanda defquery/run-query. Exemplu 2




```
(deftemplate personal-data
                (slot name)
                (slot age)
                (slot weight)
                (slot height)
                (slot processed (default FALSE)))
(deffacts people
                (personal-data (name Andrei) (age 20) (weight 80) (height 188))
                (personal-data (name Corina) (age 22) (weight 60) (height 168)))
(defquery search-by-name
                (personal-data (name ?n) (age ?a) (weight ?w) (height ?h)))
(reset)
(bind ?res (run-query search-by-name ))
(while (?res hasNext)
                (bind ?fact (call (call ?res next) fact 1))
                (bind ?n (call ?fact get 0))
                (bind ?a (call ?fact get 1))
                (bind ?w (call ?fact get 2))
                (bind ?h (call ?fact get 3))
                (printout t ?n ", age:" ?a ", weight:" ?w ", height:" ?h "." crlf))
```

Comanda defquery/run-query. Exemplul 3

```
import jess.*;
import java.util.*;

public class Main {


    public static void main(String[] args) throws JessException{
        //String pentru stocarea comenzilor Jess
        String buff;
        //cream o instanta a clasei de inferfata cu Jess
        Rete engine = new Rete();
        engine.executeCommand("(batch facts1.clp)");
        engine.reset();
        //engine.store("RESULT", engine.runQuery("search-by-name",new ValueVector()));
        engine.executeCommand("(store RESULT (run-query search-by-name))");

        Iterator e = (Iterator) engine.fetch("RESULT").externalAddressValue(null);

        while (e.hasNext()) {
            Token t = (Token) e.next();
            Fact f = t.fact(1);
            String name = f.get(0).toString();
            System.out.println(name);
        }

    }
```

```
Test Results | Output - JavaJess (run)
run:
Hello from Jess in Java application!
Andrei, age:20, weight:80, height:188.
Corina, age:22, weight:60, height:168.
Andrei
Corina
BUILD SUCCESSFUL (total time: 0 seconds)
```

Returnarea valorilor din Jess in Java (1)

Comenzile Jess rulate cu ajutorul functiei executeCommand a clasei Rete pot sa genereze valori de return (altele decat TRUE sau FALSE). In acest caz, datele de return pot fi salvate intr-un obiect de tip Value.

```
import jess.*;
....
Rete engine = new Rete();
....
Value result = engine.executeCommand("(+ 2 2)");
System.out.println(result.intValue(null));
```

Returnarea valorilor din Jess in Java (2)

```
import jess.*;
public class ExSquare
{
  public static void main(String[] unused)
  {
    try
      {
        Rete r = new Rete();
        r.executeCommand("(deffunction square (?n) (return (* ?n ?n)))");
        Value v = r.executeCommand("(square 3)");

        // Prints '9'
        System.out.println(v.intValue(r.getGlobalContext()));
      }
    catch (JessException ex)
      {
        System.err.println(ex);
      }
  }
}
C:\> java ExSquare
    9
```

Routere de Intrare/Iesire
I/O Routers

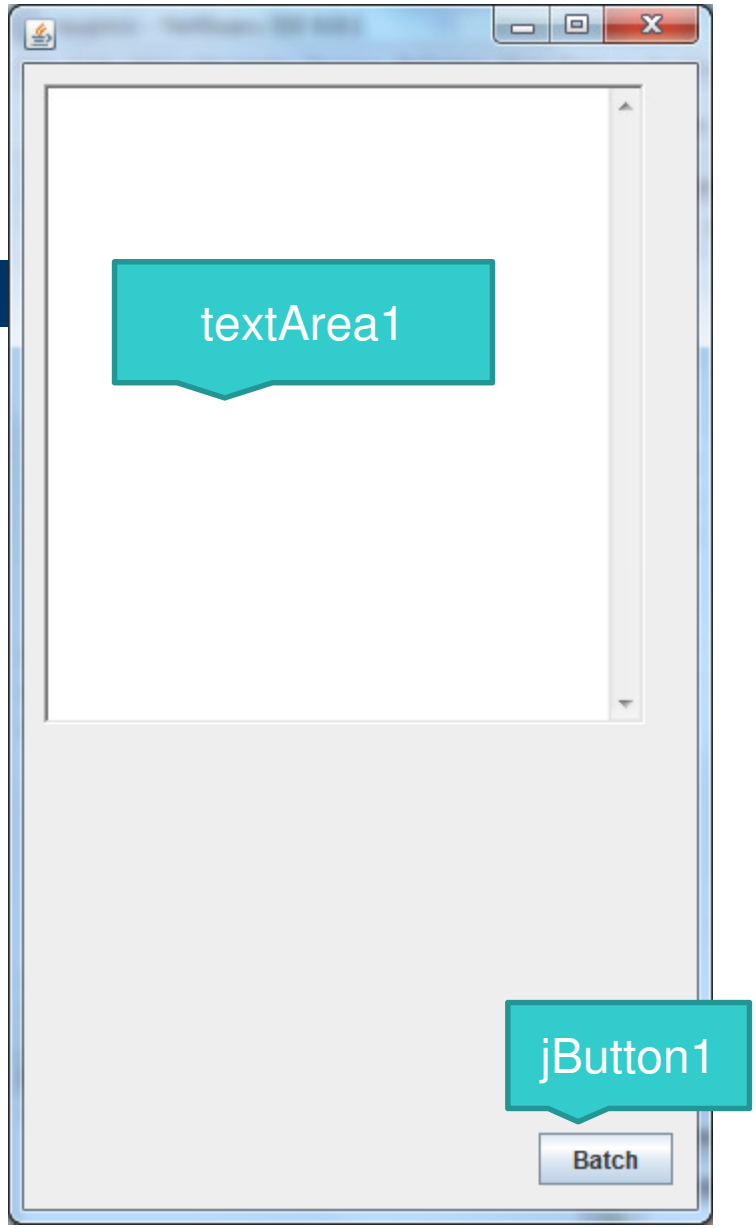
Functiile Jess printout sau read au ca parametru standard codul t ce reprezinta consola standard de iesire, respectiv de intrare (instructiunile read si readline: nu e necesar sa fiu urmate de acest cod, el fiind considerat default).

Totusi routerele standard Jess sunt connectate la streamurile standard Java, si datorita acestui lucru ele pot fi rutate la obiecte grafice GUI, dupa cum urmeaza:

- jess.awt.TextReader pentru stream-ul de intrare
- jess.awt.TextWriter pentru stream-ul de iesire

Routere de Intrare/Iesire
I/O Routers. Exemplu

Cream o fereastra Java (de tip javax.swing.JFrame) care sa aiba un obiect de tip java.awt.TextArea si un buton (javax.swing.JButton).

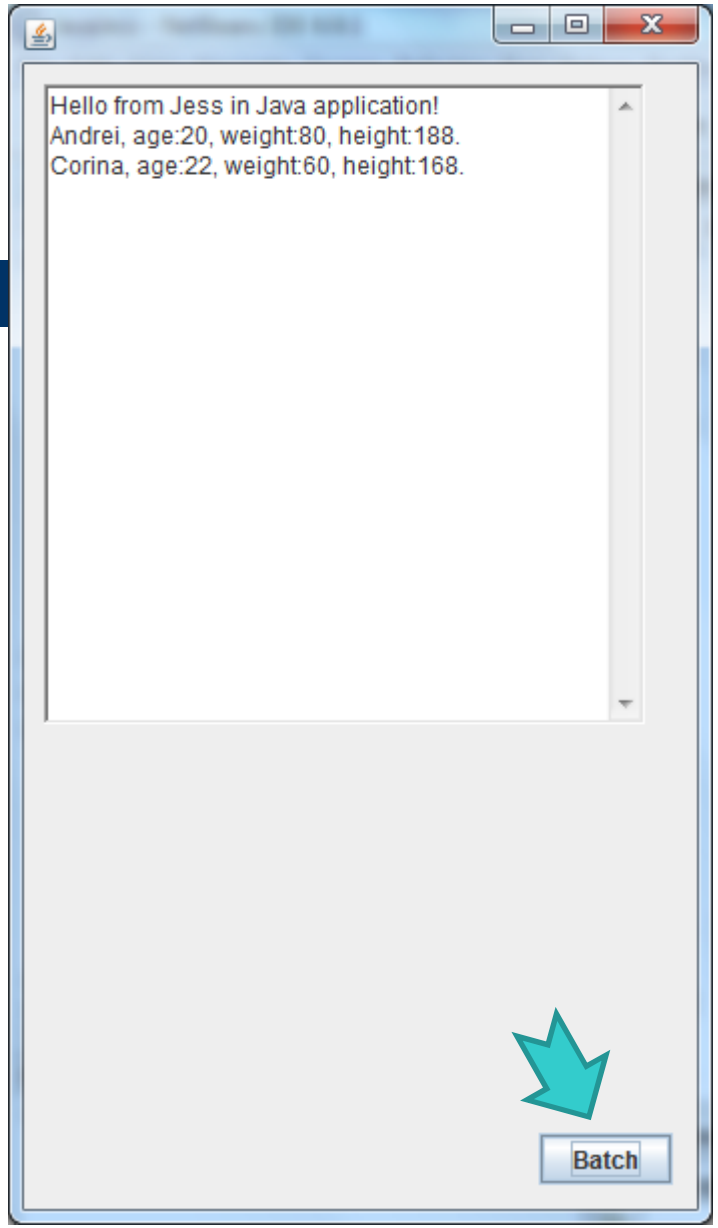


Routere de Intrare/Iesire
I/O Routers. Exemplu

Modificam functia ActionPerformed atasata butonului de pe interfata grafica astfel:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        Rete engine = new Rete();
        TextAreaWriter taw = new TextAreaWriter(textArea1);
        engine.addOutputRouter("t", taw);
        engine.addOutputRouter("WSTDOUT", taw);
        engine.addOutputRouter("WSTDERR", taw);
        try{
            engine.executeCommand("(batch facts1.clp)");
        }
        catch(JessException ex)
        {System.out.println(ex.getMessage());}
    }
```


Very
Well
Done


Hello from Jess in Java application!
Andrei, age:20, weight:80, height:188.
Corina, age:22, weight:60, height:168.
Batch

Tipuri de date Jess si corespondentele din Java

Simboluri speciale in Jess	
Nume	**Semnificatie Java**
nil	null
TRUE	java.lang.Boolean
FALSE	
crlf	\n (newline)
RU.STRING	String
RU.INTEGER	byte, short, int
RU.LONG	long
RU.FLOAT	double, float
RU.ATOM	char sau java.lang.Character
RU.LIST	un tablou Java
RU. EXTERNAL_ADDRESS	instanta a unui obiect Java

Construirea de fapte neordonate Jess in aplicatii Java

Construirea de fapte neordonate Jess dintr-o aplicatie Java se face folosind instante ale clasei jess.Fact. Orice astfel de constructie trebuie precedata de comanda executeCommand a clasei Rete pentru definirea template-ului corespunzator si trebuie urmata de asertarea in memorie a faptului, folosind functia assertFact.

```
import jess.*;
...
Rete engine = new Rete();
engine.executeCommand(”(deftemplate nume-fapt...)”);
Fapt f = new Fact(”nume-fapt”, engine);
f.setSlotValue(nume-slot, valoare);
...
engine.assertFact(f);
```

Construirea de fapte neordonate Jess in aplicatii Java. Exemplu

```
import jess.*;
public class ExPoint
{
  public static void main(String[] unused) throws JessException
  {
    Rete r = new Rete();
    r.executeCommand("(deftemplate point \"A 2D point\" (slot x) (slot y))");

    Fact f = new Fact("point", r);
    f.setSlotValue("x", new Value(37, RU.INTEGER));
    f.setSlotValue("y", new Value(49, RU.INTEGER));
    r.assertFact(f);

    r.executeCommand("(facts)");
  }
}
C:\> java ExPoint
    f-0    (MAIN::point (x 37) (y 49))
  For a total of 1 facts.
```

Construirea de fapte neordonate Jess in aplicatii Java. Definirea campurilor multislot

In Java putem construi un camp multislot prin intermediul unei instante de clasa Value pentru care precizam tipul de date RU.LIST. Datele campului multislot sunt stoate intr-un vector ValueVector, atasat obiectului Value.

```
import jess.*;
...
Rete engine = new Rete();
engine.executeCommand("(deftemplate nume-template ... (multislot values)");
Fact fact = new Fact("nume-template", engine);
ValueVector vect = new ValueVector();
vect.add(new Value("nume-valoare", RU.[tip]));
...
f.setSlotValue("values", new Value(vect, RU.LIST));
```

Construirea de fapte neordonate Jess in aplicatii Java. Definirea campurilor multislot. Exemplu

```
import jess.*;
public class ExMulti
{
  public static void main(String[] unused) throws JessException
  {
    Rete r = new Rete();
    r.executeCommand("(deftemplate vector \"A named vector\"" +
                     " (slot name) (multislot list))");

    Fact f = new Fact("vector", r);
    f.setSlotValue("name", new Value("Groceries", RU.ATOM));
    ValueVector vv = new ValueVector();
    vv.add(new Value("String Beans", RU.STRING));
    vv.add(new Value("Milk", RU.STRING));
    vv.add(new Value("Bread", RU.STRING));
    f.setSlotValue("list", new Value(vv, RU.LIST));
    r.assertFact(f);

    r.executeCommand("(facts)");
  }
}
C:\> java ExMulti
    f-0    (MAIN::vector (name Groceries) (list "String Beans" "Milk" "Bread"))
  For a total of 1 facts.
```

Construirea de fapte ordonate Jess in aplicatii Java

Un fapt ordonat este considerat drept un fapt neordonat care are un singur slot de tip multislot identificat prin numele __data.

Crearea de fapte ordonate nu implica definirea unui template – acesta se creaza automat in urma definirii unor astfel de fapte.

```
import jess.*;
...
Rete engine = new Rete();
Fact fact = new Fact(”culori”, engine);
ValueVector vect = new ValueVector();
vect.add(valoare, tip-valoare);
...
fact.setSlotValue(”__data”, new Value(vect,RU.LIST));
engine.assertFact(fact);
```

Construirea de fapte ordonate Jess in aplicatii Java. Exemplu

```
import jess.*;
public class ExOrdered
{
  public static void main(String[] unused) throws JessException
  {
    Rete r = new Rete();

    Fact f = new Fact("letters", r);
    ValueVector vv = new ValueVector();
    vv.add(new Value("a", RU.ATOM));
    vv.add(new Value("b", RU.ATOM));
    vv.add(new Value("c", RU.ATOM));
    f.setSlotValue("__data", new Value(vv, RU.LIST));
    r.assertFact(f);

    r.executeCommand("(facts)");
  }
}
```

Clasa jess.Deftemplate

Definirea template-urilor pentru fapte Jess se poate realiza din Java prin intermediul unei instante a clasei jess.Deftemplate.

```
import jess.*;
...
Rete engine = new Rete();
Deftemplate dt = new Deftemplate(”nume-templ”, ”comentariu”, engine);
dt.addSlot(”nume-slot”, new Value(valoare, RU.[tip]), ”tip-jess”);
engine.addDeftemplate(dt);
```

Clasa jess.Deftemplate. Exemplu

```
import jess.*;
public class ExBuildDeftemplate
{
  public static void main(String[] unused) throws JessException
  {
    Rete r = new Rete();
    Deftemplate dt = new Deftemplate("point", "A 2D point", r);
    Value zero = new Value(0, RU.INTEGER);
    dt.addSlot("x", zero, "NUMBER");
    dt.addSlot("y", zero, "NUMBER");
    r.addDeftemplate(dt);

    // Now create and assert Fact
  }
}
```