1 Preparation of environment

Before everything else please open our project in your Java Integrated Development Environment:

1.1 Available instructions:



1.1.1 Configuration in IntelliJ:

This tutorial step by step is how to import existing eclipse's project to IntelliJ, so if You know how to import, skip until section <u>Exercise 0</u>.

1. Let's run IntelliJ:



2. Let's unpack our package

_

3. Import Project



a. If somebody does not see that window importing can be selected: File->New->Project from Existing Sources:

13	JADE -	exercis	se_student	s_1_Inte	IIIJ [E:\Dro	pbox\DO		NCKIE	\jade_2	201/na	a18\Jade	e_er	lg\JAL
Eil	e <u>E</u> dit	<u>V</u> iew	<u>N</u> avigate	<u>C</u> ode	Analy <u>z</u> e	<u>R</u> efactor	<u>B</u> uild	R <u>u</u> n	<u>T</u> ools	VC <u>S</u>	<u>W</u> indo	W	<u>H</u> elp
					×	<u>P</u> rojec	t						
	<mark>7</mark> <u>O</u> pen.					Projec	t from	Existi	ng Sou	rces			
	Open <u>R</u> ecent ►					Projec	t from	Versi	on Con	trol	►		
	Close	Project	t			<u>M</u> odu	le						
۶	Se <u>t</u> ting	gs		Ct	rl+Alt+S	Modu	le fron	n Exist	ing Sou	urces			
	Projec	t Struc	ture	Ctrl+Alt∙	+Shift+S	🛃 Scrato	h File	Ctr	l+Alt+S	Shift+1	Insert		
	Other	Setting	gs		Þ								

4. Choose path to unpacked project (it is possible to paste path to project), then OK:



5. Choose "Import project from external model" and choose Eclipse, then Next:



6. In window "Select Eclipse projects directory" You can press just Next, or change something.

7. You should see project:

Project	and the banks	allie Manufactor	X
Select Eclipse projects to import			
✓ JADE - exercise			
		Select <u>a</u> ll	<u>U</u> nselect all
Open Project Structure after import			
	Previous	<u>N</u> ext Cancel	Help

select "Open Project Structure after import", then Next

- 8. In window "Choose project code style" You can press just Next, or change something.
- 9. In next window You probably will see possibility with selecting JDK, just press Finish.

1.1.2 Running application in IntelliJ:

1. Choose Run -> Edit Configurations...



2. In a new window press +, then choose Application, and type in Main class and Program arguments what You see below:

🖳 Run - Run Jade						
+ - 恒 🖌 🔺 🖛 »	Name: Run Jade	Single instance only				
Application	Configuration Code (
🔚 Run Jade	Configuration					
F Templates	Main <u>c</u> lass:	jade.Boot				
	<u>V</u> M options:					
	Working directory:	E:\Dropbox\DOKTORANCKIE\jade_2017na18\Jade_eng\JADE - exercise_students_1_IntelliJ				
	Environment variables:					
	Use classpath of module:	JADE - exercise				
		Include dependencies with "Provided" scope				
		Default (1.8 - SDK of 'JADE - exercise' module)				
	Shorten command <u>l</u> ine:	user-local default: none - java [options] classname [args]				
	Enable capturing form					
	▼ <u>B</u> efore launch: Build, Activa	te tool window				
	+ - /					
	🔨 Build					
	Show this page 🗹 A	ctivate tool window				
?		Cancel Apply				

3. Press Run.

- 4. When we will have problems with port we would have to use another port with argument: host 192.168.2.9 -port 12344
- 5. In other problems it can help if we download newest version of JADE framework: http://jade.tilab.com/download/jade/

1.1.3 Configuration in Eclipse:

1. Let's run Eclipse:



- 2. Let's unpack our package
- 3. File ->Import



4. General -> Existing Project into Workspace -> Next:

E Import	
Select Create new projects from an archive file or directory.	Ľ
Select an import wizard:	
type filter text	
 ✓ General ✓ Archive File ✓ Existing Projects into Workspace ✓ File System ✓ Preferences ✓ Projects from Folder or Archive ✓ Git ✓ Gradle ✓ Install ✓ Maven ✓ Oomph ✓ Run/Debug ✓ Tasks ✓ Team × XMI 	
(?) < <u>Back</u> <u>Next ></u> <u>Finish</u>	Cancel

5. Browse button, in which we have to choose path of our unpacked archive and project, after that press Finish:

Import		
Import Projects Select a directory to sear	ch for existing Eclipse projects.	
 Select roo<u>t</u> directory: Select <u>a</u>rchive file: <u>P</u>rojects: 	C:\Users\Myśko i Grześko\Desktop\jade_rozpakowane\JADE - exercise	Browse
JADE - cwiczenia	(C:\Users\Myśko i Grześko\Desktop\jade_rozpakowane\JADE - exercise)	Select All
Options Searc <u>h</u> for nested pro Copy projects into w Hide projects that alr Working sets Add projec <u>t</u> to work W <u>o</u> rking sets:	jects orkspace eady exist in the workspace ing sets	Ne <u>w</u> S <u>e</u> lect
?	< <u>B</u> ack <u>N</u> ext > <u>Finish</u>	Cancel

6. After that we should see project with four packages:

🖨 e	🖨 eclipse-workspace - Eclipse										
<u>F</u> ile	Edit	Source	Refactor	<u>N</u> avigate	Se <u>a</u> rch	<u>P</u> roject	<u>R</u> un	Window	Help		
: =0	- m	100: -	- 0 -	0 - 0	: ==>	a - 1 m	10	- 🙈 : 🔿			
	Í	18 : 🎌		~ • ~	• : #	G • : 🗁	<u> </u>	• 🗁 : 🝽			
🛱 I	📱 Package Explorer 🔀 📄 😤 📮 🗖										
4	a 😸 JADE - cwiczenia										
	⊿ 🕭	src									
	kis.sspd.jade.exercise0										
		> 🚺 H	elloWorld.ja	ava							
		> 🕽 H	elloWorld2	java							
	⊿	🖶 kis.ss	pd.jade.exe	rcise1							
		For Provide the	orgetful.java	э							
			stener.java								
			alker.java								
	4	KIS.SS KIS.SS KIS.SS	pa.jade.exe	rcisez							
			ng.java ingDong iau	(a							
			ngrong.jav	a							
	4	🖶 kis.ssi	od.iade.farr	n							
	_	⊳ 🗊 Br	reeder.iava								
		5 🚺 Bu	unnyFemal	e.java							
		> 🚺 Bu	unnyMale.ja	ava							
		> 🕽 W	/olf.java								
	⊳ 🛋	JRE Syste	m Library [J	lavaSE-1.7]							
	⊳ 🛋	Reference	ed Libraries								
	4 🗁	lib									
	\triangleright	🔁 comn	nons-code	C							
		APDe	scription.tx	t							
	1	jade.j	ar								
	~	MIPs	-Main-Con	itainer.txt							
	▲ 🥏	lib_old	none codo	-							
	Þ		scription ty	+							
		📄 iade i	ar								
		MTPs	 :-Main-Con	tainer.txt							
		APDescri	ption.txt								
		MTPs-Ma	ain-Contair	ner.txt							

2.1.1 Running application in Eclipse

Let's try to run project, if you see any problems please check:

- 1. if all libraries are properly added:
 - a. Project ->Properties



b. Java Build Path ->Libraries:



- c. If we don't have those libraries we have to add them manually
- 2. Our running configuration and program arguments are correct:

a. Run ->Run Configurations ...

🖨 eclipse-workspace - Eclipse			
File Edit Source Refactor Navigate Search Project	Run	Window Help	
: ┌; - : : : : : : : : : : : : : : : : : :	Q	Run	Ctrl+F11
	橋	Debug	F11
Package Explorer 🛛 🕒 😒 🜍 🗸 🗠	-	Coverage Last Launched	Ctrl+Shift+F11
A 😂 JADE - cwiczenia	1		
▲ ([™]) src		Run History	•
Kis.sspd.jade.exercise0		Run As	•
HelloWorld2.iava		Run Configurations	
kis.sspd.jade.exercise1		Debug History	•
Forgetful.java		Debug As	•
D Listener.java		Debug Configurations	
D Talker.java		Debug Conngulations	
kis.sspd.jade.exercise2		Coverage History	+
Ping-Java		Coverage As	•
Pingrong.java Pongriava		Coverage	
 kis.sspd.jade.farm 	_	Tagala Praskusint	Chally Childry D
D Breeder.java			Ctri+Shiit+B
D BunnyFemale.java	0	loggle l'racepoint	
BunnyMale.java	0	Toggle Line Breakpoint	
Wolf.java	000	Toggle Watchpoint	
JRE System Library [JavaSE-1./]	0	Toggle Method Breakpoint	
Referenced Libraries	à	Skip All Breakpoints	Ctrl+Alt+B
Commons-codec	Sec.	Remove All Breakpoints	
APDescription.txt	٦Û	Add Java Exception Breakpoint	
🙀 jade.jar	Θ	Add Class Load Breakpoint	
MTPs-Main-Container.txt	_	AU.D. (
⊿ 🗁 lib_old	12	All References	
Commons-codec	0	All Instances	Ctrl+Shift+N
iade iar		Instance Count	
MTPs-Main-Container.txt	x+y =?	Watch	
APDescription.txt	Q	Inspect	Ctrl+Shift+I
MTPs-Main-Container.txt	U)	Display	Ctrl+Shift+D
	Qj	Execute	Ctrl+U
		Force Return	Alt+Shift+F
	%	External Tools	•

b. *Java Application ->New*, main class should be **jade.Boot**

Run Configurations	×
Create, manage, and run con Run a Java application	figurations
📑 🖹 🗖 🗱	Name: New_configuration
type filter text	🞯 Main 🖉 Arguments 🛋 JRE 🔩 Classpath 🦻 Source 🖾 Environment 🔲 Common
Gradle Project	Project:
Java Applet	JADE - cwiczenia Browse
New_configuration	Main class:
Ju JUnit	jade.Boot
m2 Maven Build	Includ <u>e</u> system libraries when searching for a main class
Ju Task Context Test	Include inherited mains when searching for a main class
	Stop in main
Filter matched 8 of 8 items	Re <u>v</u> ert Apply
?	Run Close

c. Arguments -> Program arguments, we need to add **-gui**:

Create, manage, and run con	figurations
Image: Second system Image: Second system <th>Name: New_configuration Image: Arguments Image: Program arguments: -guil VM arguments: VM arguments: VM arguments: VM arguments: VM arguments: VM arguments: VM arguments:</th>	Name: New_configuration Image: Arguments Image: Program arguments: -guil VM arguments: VM arguments: VM arguments: VM arguments: VM arguments: VM arguments: VM arguments:
Filter matched 8 of 8 items	

- d. We can also set some name different than New_configuration
- e. When we will have problems with port we would have to use another port with argument: **-host 192.168.2.9 -port 12344**
- 3. In other problems it can help if we download newest version of JADE framework: http://jade.tilab.com/download/jade/

2 Exercise 0

2.1 The JADE Environment

What is **JADE**? JADE (Java Agent DEvelopment Framework) is a software Framework fully implemented in the Java language. It simplifies the implementation of multi-agent systems through a middle-ware that complies with the <u>FIPA specifications</u> and through a set of <u>graphical tools</u> that support the debugging and deployment phases. It includes:

- 1. A runtime environment where JADE agents can "live" and that must be active on a given host before one or more agents can be executed on that host.
- 2. A library of classes that programmers have to/can use (directly or by specializing them) to develop their agents.
- 3. A suite of graphical tools that allows administrating and monitoring the activity of running agents.

2.1.1 JADE GUI:

If everything is OK program's named RMA window should appear, after full expanding of *AgentPlatforms* we will see "*Main-Container*" directory:

Tma@192.168.0.149:1099/JADE - JADE Remote Agent Ma					
File Actions Tools Remote Platforms Help					
	2		s 🐞 👫		Jaac
P- ☐ AgentPlatforms	1000	name	addresses	state	owner
	Sec. 1	NAME	ADDRESSES	STATE	OWNER
🕈 🧰 Main-Container	and and a				
- 📴 ams@192.168.0.149:1099/JADE	1000				
- 🙂 df@192.168.0.149:1099/JADE	and and a				
🗆 🔛 rma@192.168.0.149:1099/JADE	and and a				
	and and				
	and and				
	1000				
	1000				
	and and				
	1000				
	1000				
	and a				

• For single JVM is connected single agent platform by default. Newly added agents are added to already existing platform. At the picture below we can see Containers and Platforms:



• RMA is not agent platform. In order to close JADE (e.g. for recompiling purpose) in RMA we have to click *File ->Shut down Agent Platform*. If we close RMA without closing platform we have to manually kill processes from Eclipse.

1 rma@192.168.0.149:1099/JADE - JADE Remote Ager	t Management GUI		- • ×
File Actions Tools Remote Platforms Help			
Close RMA Agent 🔋 📴 🚳 🗿 🚺) 🛐 🛛 🍰 😫 🛔	5	Jaac
Exit this Container	name addres	sses state	owner
192.100.0.149.1099/JADE"	ams@192.168 http://Pofe	Ikowy: active	NONE
 Main-Container ams@192.168.0.149:1099/JADE df@192.168.0.149:1099/JADE rma@192.168.0.149:1099/JADE 	E		

- Each running instance of the JADE runtime environment is called a Container as it can contain several agents. The set of active containers is called a Platform. A single special *Main container* must always be active in a platform and all other containers register with it as soon as they start. It follows that the first container to start in a platform must be a main container while all other containers must be "normal" (i.e. non-main) containers and must "be told" where to find (host and port) their main container (i.e. the main container to register with).
- After running the platform we can see 3 already running agents:
 - The **RMA** (Remote Monitoring Agent) allows controlling the life cycle of the agent platform and of all the registered agents. The distributed architecture of JADE allows also remote controlling, where the GUI is used to control the execution of agents and their life cycle from a remote host.
 - The **AMS** (Agent Management System) that provides the naming service (i.e. ensures that each agent in the platform has a unique name) and represents the authority in the platform (for instance it is possible to create/kill agents on remote containers by requesting that to the AMS).
 - The **DF** (Directory Facilitator) that provides a Yellow Pages service by means of which an agent can find other agents providing the services he requires in order to achieve his goals.



Please spend few minutes to explore RMA, because we are going to use its during rest of the day.

2.2 Hello word

After selecting "Main-Container" please clik Actions ->Start New Agents in order to create new agent:

🔋 m	🚽 rma@192.168.0.149:1099/JADE - JADE Remote Agent Management GUI										
File	Actions Tools Remo	te Platforms Help									
C	Start New Agent	9 9 6 6	🍰 📴 🌛 👹			Uaae					
۰ 🗅	🖽 Kill		name	addresses	state	owner					
	Suspend	099/JADE"									
	😅 Resume	r									
	🚇 Send Message	168.0.149:1099/JADE									
	😅 Migrate Agent	168 0 149 1099/JADE									
	Clone Agent										
	🖾 Load Agent										
	Save Agent										
	Freeze Agent										
	🔯 Thaw Agent										
Ľ		<u></u>									

Then we have to type Agent name and choose agent's class name, in order to choose class we should click ... then select *kis.sspd.jade.exercise0.HelloWorld*, after that we can create the agent by clicking *OK*.

- What happened?
- What the agent is doing?
- What can we see in console?
- Let's kill our running agent (*Actions ->Kill*), what can we observe?
- Is it possible to kill one of 3 default agents?
- Is it possible to kill main container?
- Can we create multiple agents with the same name?
- Let's look to the HelloWorld agent's code.

2.3 Hello word2

Please run agent HelloWorld2

- What do we see in console just after creating the agent? What do we see after few seconds?
- Let's try to kill the agent manually
- Let's look to the HelloWorld agent's code.

Class Agent and all derived classes inherit few methods, e.g.:

Method	Description	Return value
setup()	This protected method is an empty placeholder for application specific startup code.	void
takeDown()	This protected method is an empty placeholder for application specific cleanup code.	void
doDelete()	Make a state transition from <i>active</i> ,	void

	<i>suspended</i> or <i>waiting</i> to <i>deleted</i> state within Agent Platform Life Cycle, thereby destroying the agent.	
addBehaviour(Behaviour b)	This method adds a new behaviour to the	void
	agent.	

More information in reference: <u>http://jade.tilab.com/doc/api/index.html</u>

3. Exercise 1

3.1 Agent behaviour and lifecycle - Forgetful

The actual job an agent has to do is typically carried out within "behaviours". A behaviour represents a task that an agent can carry out and is implemented as an object of a class that extends *jade.core.behaviours.Behaviour*

Each class extending Behaviour must implement the **action()** method, that actually defines the operations to be performed when the behaviour is in execution and the **done()** method (returns a boolean value), that specifies whether or not a behaviour has completed and have to be removed from the pool of behaviours an agent is carrying out.

Method	Description	Return value
action()	Runs the behaviour.	abstract void
done()	Check if this behaviour is done.	abstract boolean



- Agent lives until the method **doDelete()** is not called, even if pool of behaviours is empty.
- The only behaviour, which would return **true** by function **done()** are removed from the pool

3.1.2 The Forgetful agent

Let's look to the agent's code.

- What can be returned by function **done()** in **OneShotBehaviour**? When it is going to be removed from behaviour pool?
- What can be returned by function **done()** in **CyclicBehaviour**? When it is going to be removed from behaviour pool?

- What does it means for those behaviours?
- What would be the output of the agent in your opinion?
- Let's run the agent and check
- Uncomment the line myAgent.addBehaviour(new SingleBehaviour()); in class CyclicBehaviour2
- Comment myAgent.addBehaviour(new SingleBehaviour()); in class
 CyclicBehaviour1
- What is the output?

UML diagram of types of defined behaviours in JADE:



3.2 The Listener agent

- Run the Listener Agent
- Send do the running agent message: Actions ->Send Message
- Check the processor usage
- Add one more listener agent and check usage now

- Check usage after killing them both softly
- Add the code below after the if-statement:

else {
 block();
}

- How does the CPU usage looks now?
- Let's add to the else-statement at the end line: System.out.println("Blocked.");
- When exactly the agent is being blocked?

3.3 The Talker agent

- Run the Listener agent
- Run the Talker agent adding in field *Arguments* name of the running Listener agent.
- Let's take a look into code of both agents. What types of arguments are being sent to the Listener agent

3.4 Way of communication

One of the most important features that JADE agents provide is the ability to communicate and interactions protocols. The communication paradigm adopted is the asynchronous message passing. Each agent has a sort of mailbox (the agent message queue) where the JADE runtime posts messages sent by other agents. Whenever a message is posted in the message queue the receiving agent is notified. If and when the agent actually picks up the message from the message queue to process it is completely up to the programmer however. All received messages are being hold in FIFO queue, but it is available mechanism to search in the queue with patterns. Messages are objects containing lots of information and having many methods to help communicate (more details soon).



The JADE asynchronous message passing paradigm

4 Exercise 2

- Run the PingPong agent
- Take a look to the source code of all tree agents: Ping, Pong, PingPong
- How they communicate?
- How new agents are being created? Who is their creator?
- What would happen (or rather not) if we don't call the **start()** method

Bibliography:

- JADE home page: <u>http://jade.tilab.com</u>
- JADE official tutorial from which many sentences are copied: http://jade.tilab.com/doc/tutorials/JADEProgramming-Tutorial-for-beginners.pdf
- Another JADE tutorial, which added some information to the document: <u>http://jade.tilab.com/doc/administratorsguide.pdf</u>
- Robert's tutorial in polish