



THE ICASA PLATFORM

PHILIPPE LALANDA / COLIN AYGALINC
KOBE UNIVERSITY – AUGUST 2017

STRUCTURE

Generality about iCasa

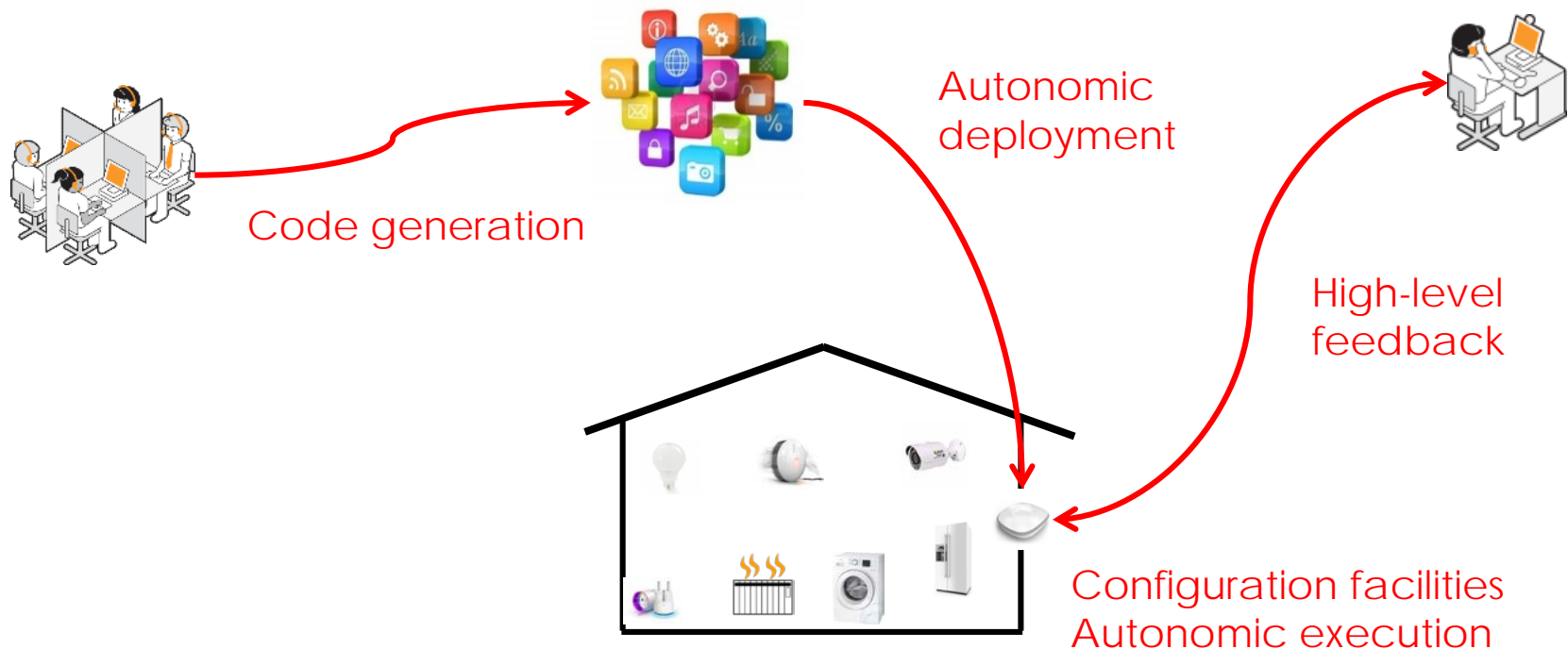
Details

Expected exercises

ICASA

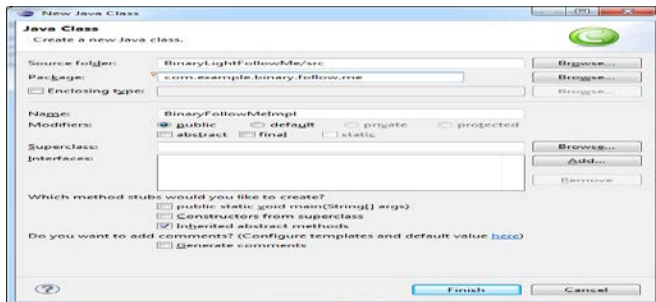
A software environment for the development, deployment, configuration, execution of pervasive applications:

- Simplifying these tasks



ICASA OVERVIEW

Eclipse-based environment



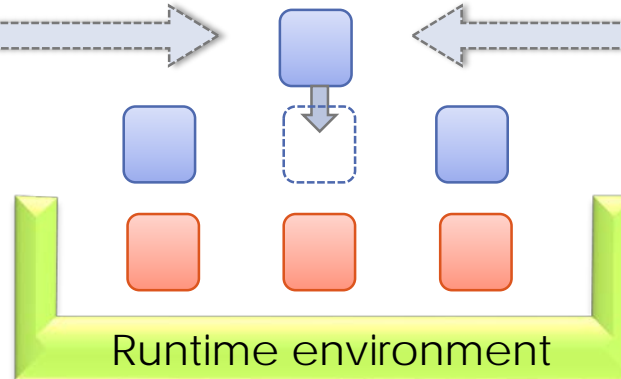
Configuration/administration



Deployment

Synchronization

Dynamic
Pervasive app

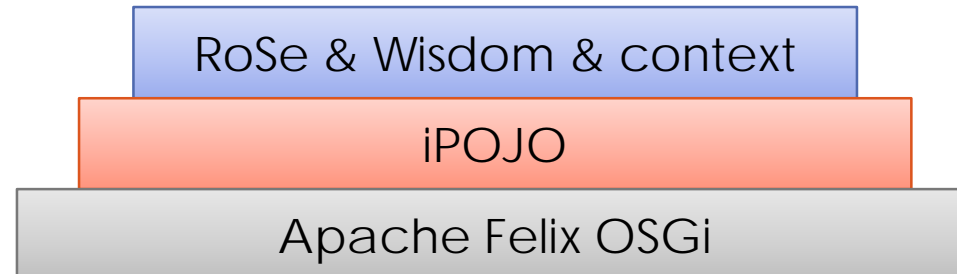


ICASA RUNTIME

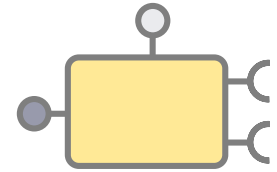
The runtime is made of several frameworks:

- RoSe (resource integration)
- iPOJO (development model)
- Wisdom (web server)
- Context management
- HMI management

Dynamic
Autonomic

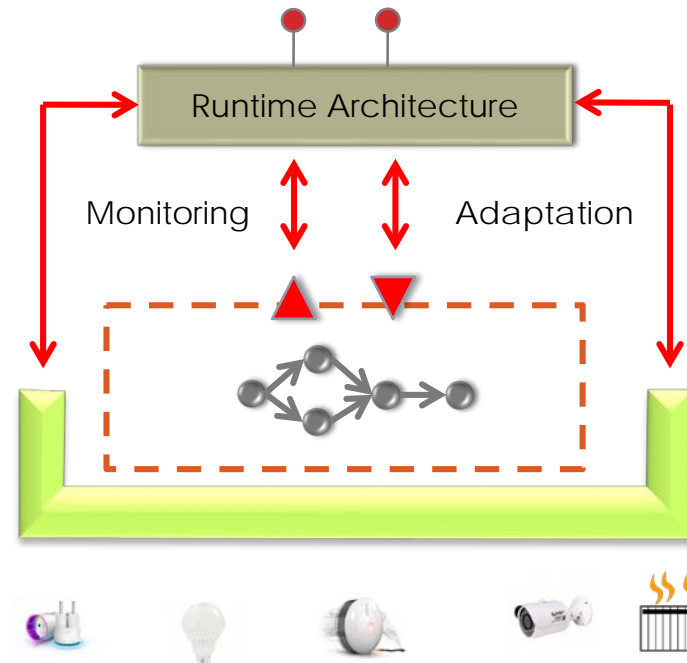


IPOJO



A service-oriented component model:

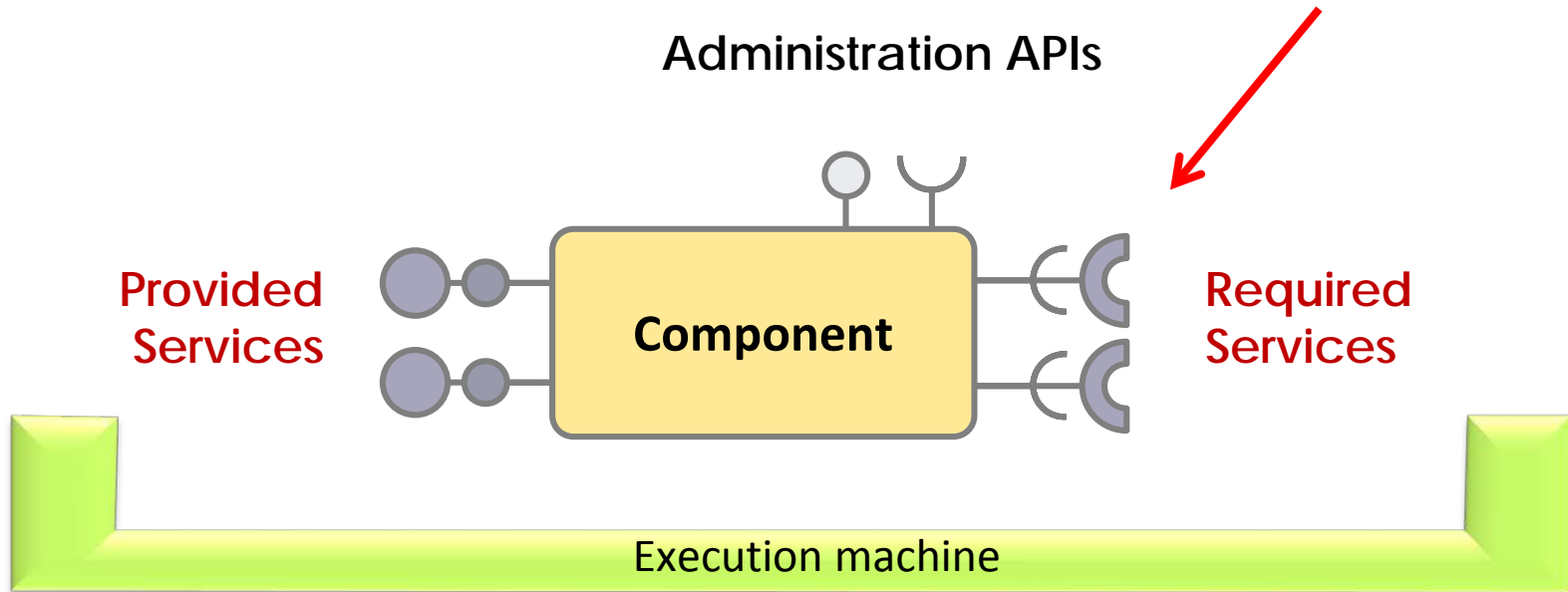
- Required and provided services (**specifications**)
- Dependencies resolved at runtime
- Re-assessed at every context modification
- Built-in runtime architecture



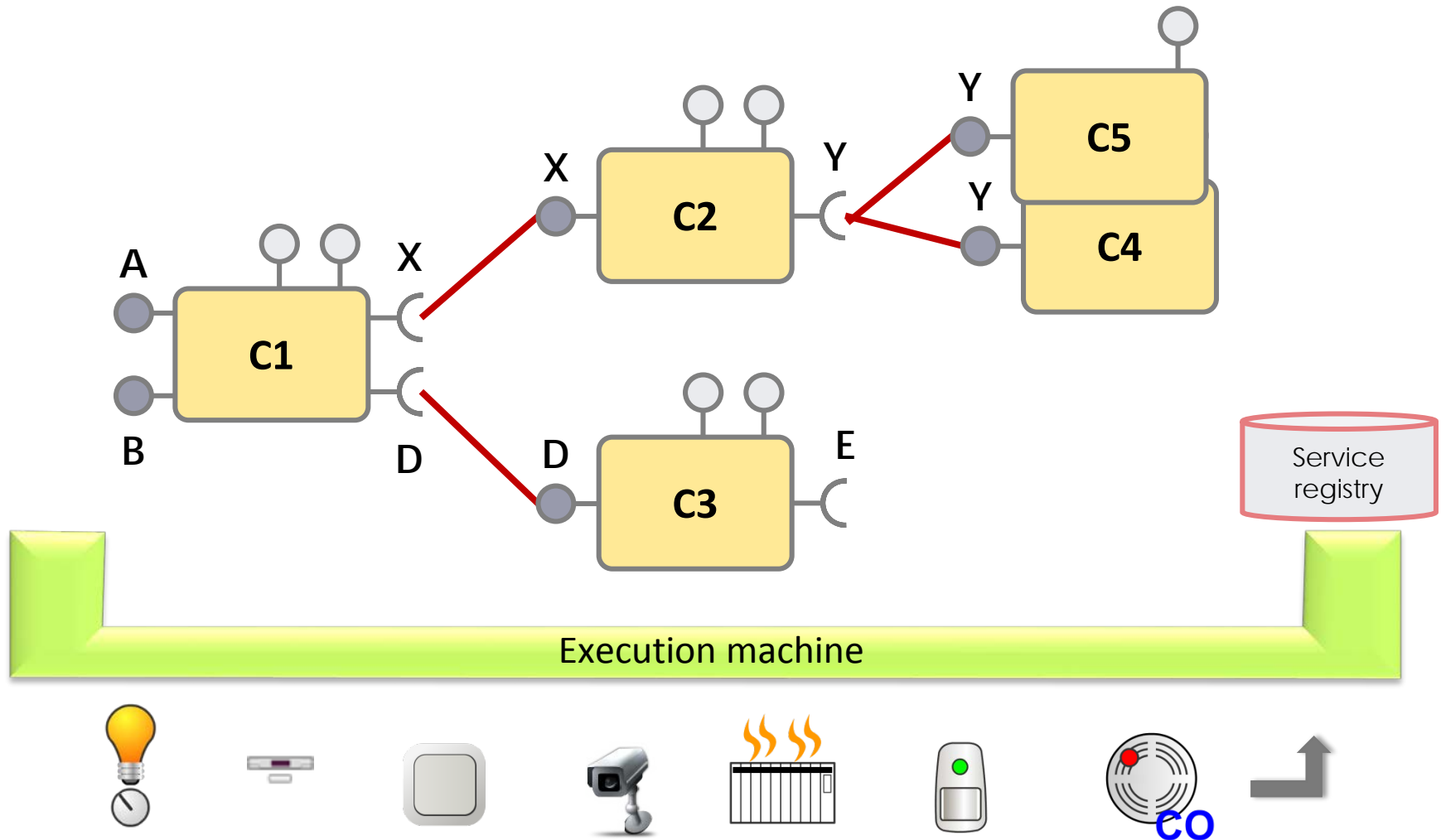
Open Source (Apache)

IPOJO

A service-oriented component model with self-adaptation features.



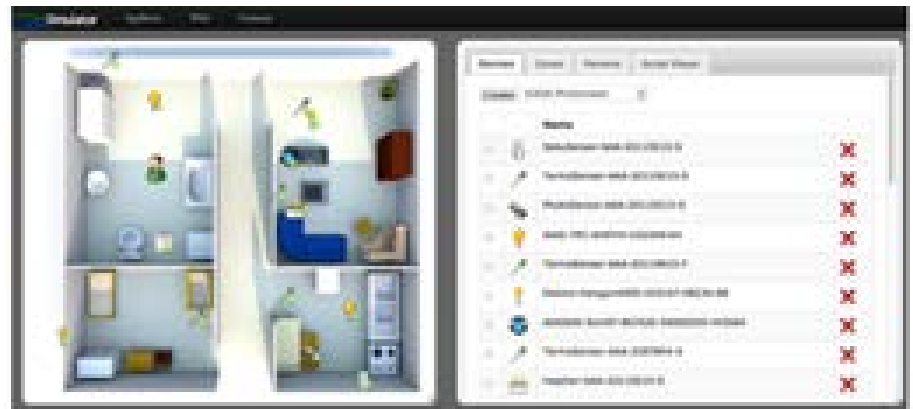
IPOJO - ILLUSTRATION



WEB-BASED CONFIGURATION/TEST

Represents the environment, including a map of the house

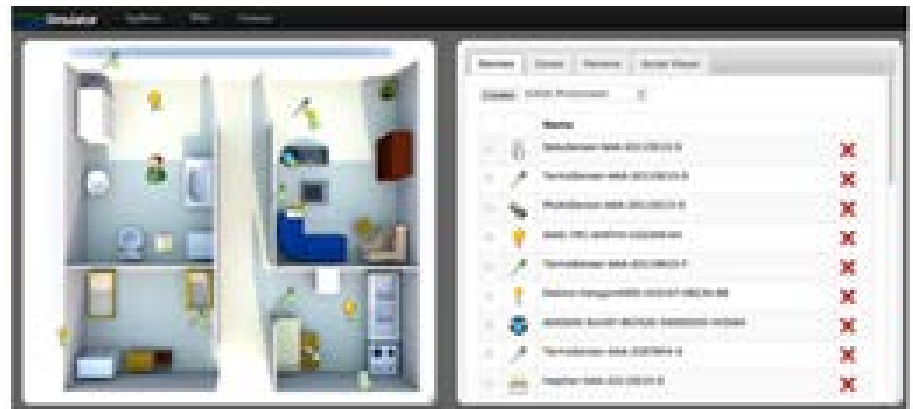
- Full control over time, devices, zone, simulated persons
- Mixes simulated and real devices
- Synchronized
- Extensible



CURRENT WORK

Better context

- More than “just devices”
- Abstract services
- In line with application needs



STRUCTURE

Generality about iCasa

Details

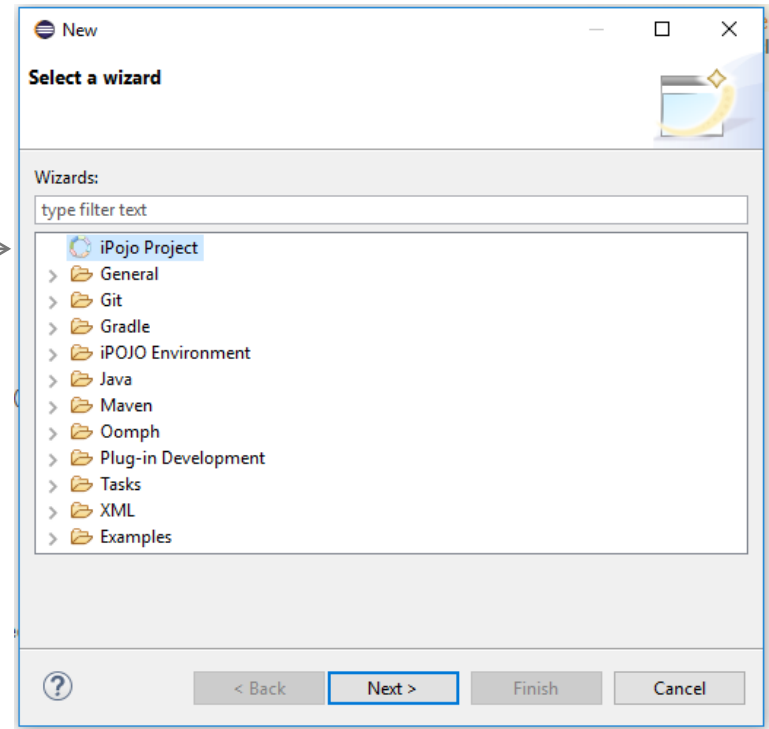
Expected exercises

ECLIPSE ENVIRONMENT

Toolbox integrated as an Eclipse plug in

- How to install it ? http://self-star.imag.fr/?page_id=243
- How to use it ?

Select →

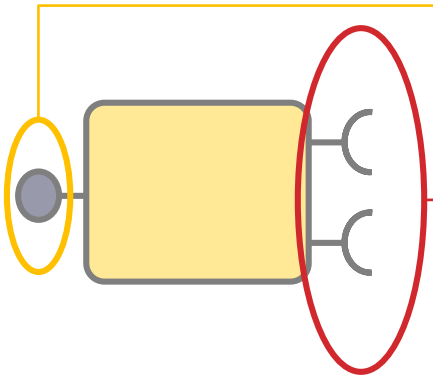


ECLIPSE ENVIRONMENT

Use it to describe your iPOJO component

Services that your component provides

Services that your component requires to work



The diagram shows a yellow rectangular component. On its left side, there is a small grey circle with a horizontal line extending from it, enclosed in a yellow oval. An arrow points from this oval to the 'org.service.ServiceProvidedContract' entry in the 'Provided Services Specifications' section of the iPOJO Metadata Editor. On the right side of the component, there are two semi-circular connectors, each with a horizontal line extending from it, enclosed in a red oval. An arrow points from this oval to the 'Required Services' table in the iPOJO Metadata Editor.

iPOJO Metadata Editor

Component Type Definition

Bundle Component Types List

Bundle component type definition, details are editable on the right

MyFirstComponent

Add

Delete

Component Details

Set the description of the selected component.

Component Name: MyFirstComponent

Provided Services Specifications

Set the interfaces of the selected component.

org.service.ServiceProvidedContract

Add

Edit

Delete

Required Services

Set the service dependencies of the selected component.

Id	Cardinality	Type	
serviceRequired	1..1	org.service.ServiceRequiredContract	Add
servicePrime	0..1	org.service.ServiceRequiredContractPrime	Edit
			Delete

Component Properties

Set the properties of the selected component.

Name	Field	Type	Value	
				Add
				Edit
				Delete

Component Lifecycle Callbacks

Set Component lifecycle callbacks of selected component.

validate invalidate

Component Type Implementation

Set or generate the component type implementation class.

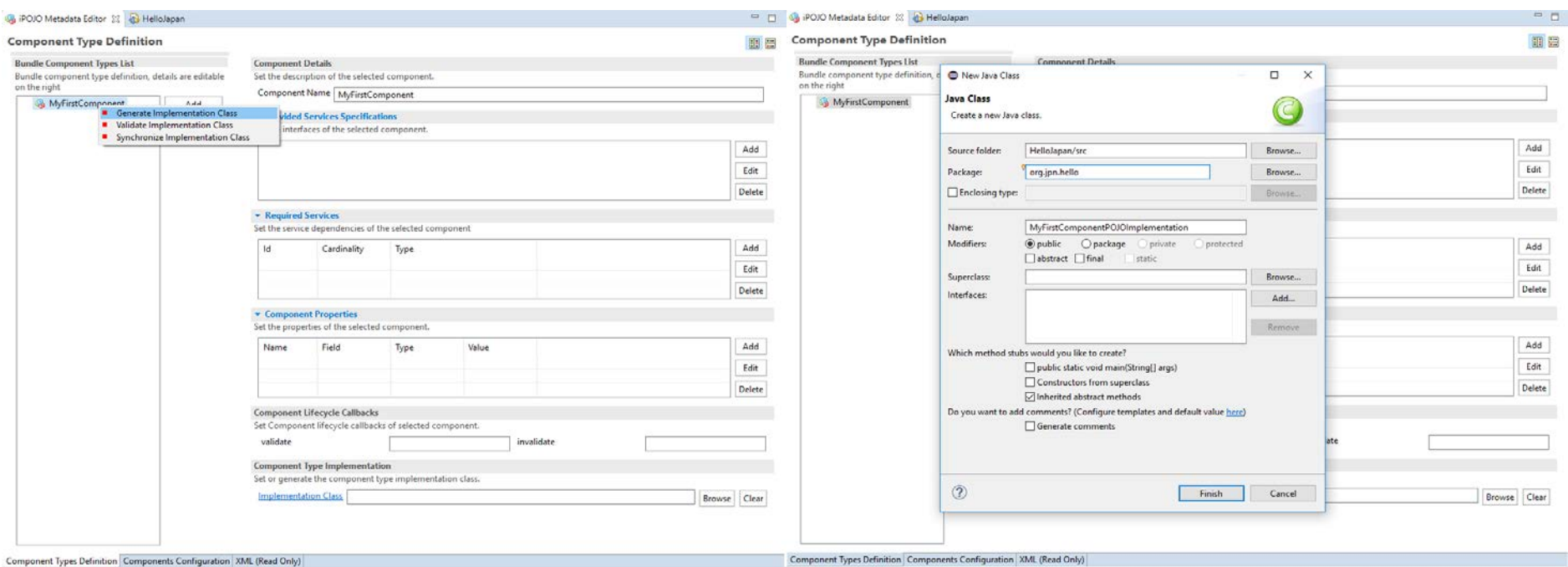
Implementation Class Browse Clear

Component Types Definition Components Configuration XML (Read Only)

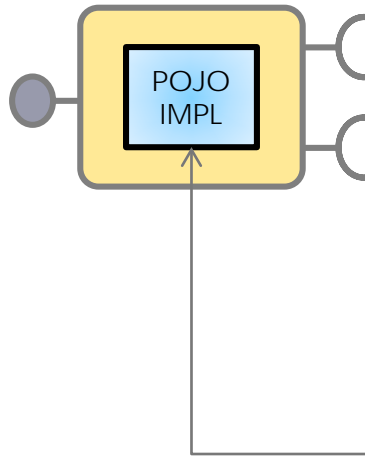
ECLIPSE ENVIRONMENT

Generates the implementation class:

- It will be the place where you code your application/service.



ECLIPSE ENVIRONMENT



iPOJO Metadata Editor | HelloJapan | ServiceProvidedContract.java | ServiceRequiredContract.java | MyFirstComponentPOJOImpl.java

Component Type Definition

Bundle Component Types List
Bundle component type definition, details are editable on the right

MyFirstComponent [Add] [Delete]

Component Details
Set the description of the selected component.
Component Name: MyFirstComponent

Provided Services Specifications
Set the interfaces of the selected component.
org.service.ServiceProvidedContract [Add] [Edit] [Delete]

Required Services
Set the service dependencies of the selected component.

Id	Cardinality	Type	
serviceRequired	1..1	org.service.ServiceRequiredContract	[Add] [Edit] [Delete]
servicePrime	0..1	org.service.ServiceRequiredContractPrime	

Component Properties
Set the properties of the selected component.

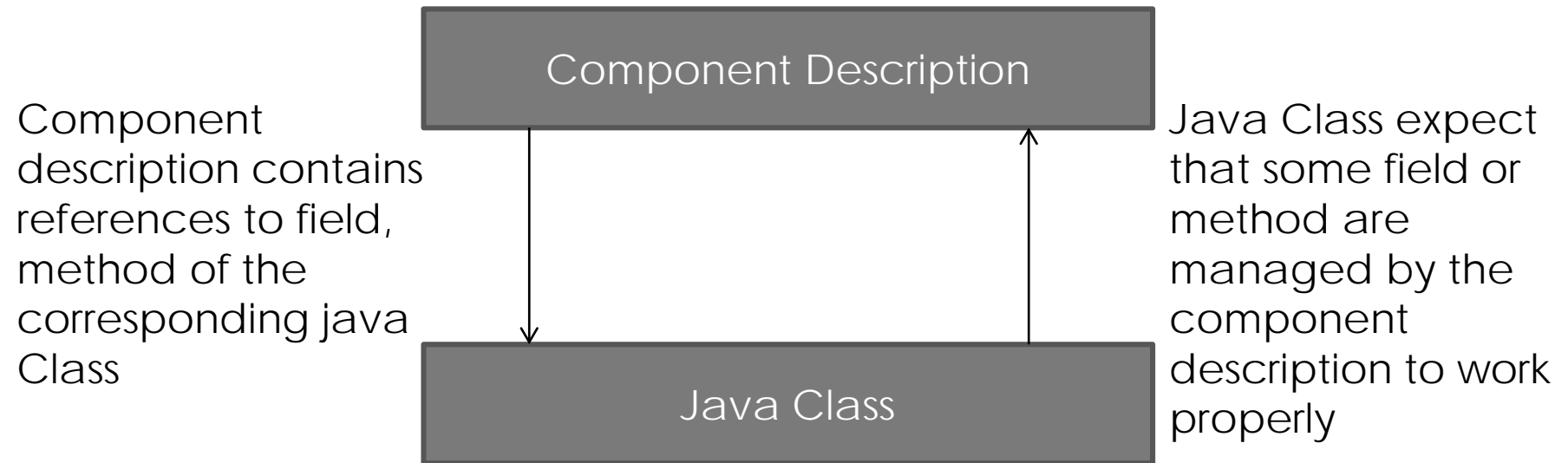
Name	Field	Type	Value	
				[Add] [Edit] [Delete]

Component Lifecycle Callbacks
Set Component lifecycle callbacks of selected component.
validate [] invalidate []

Component Type Implementation
Set or generate the component type implementation class.
Implementation Class: org.jpnp.hello.MyFirstComponentPOJOImpl [Browse] [Clear]

Component Types Definition | Components Configuration | XML (Read Only)

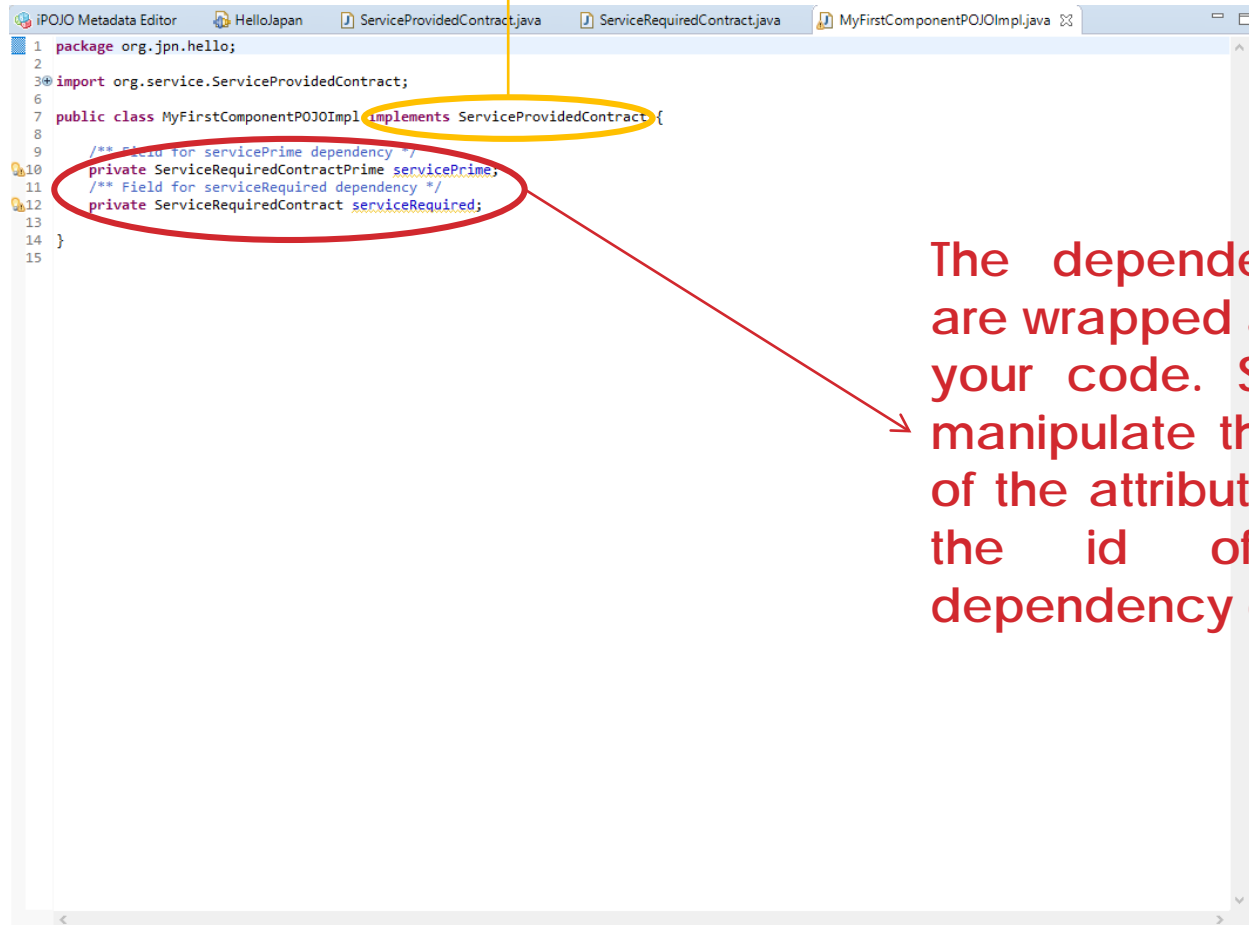
ECLIPSE ENVIRONMENT



Each change on the java Class or the component description impact the other. **Try to keep them synchronized.**

ECLIPSE ENVIRONMENT

Your implementation class must implements the service contract exposed by your component

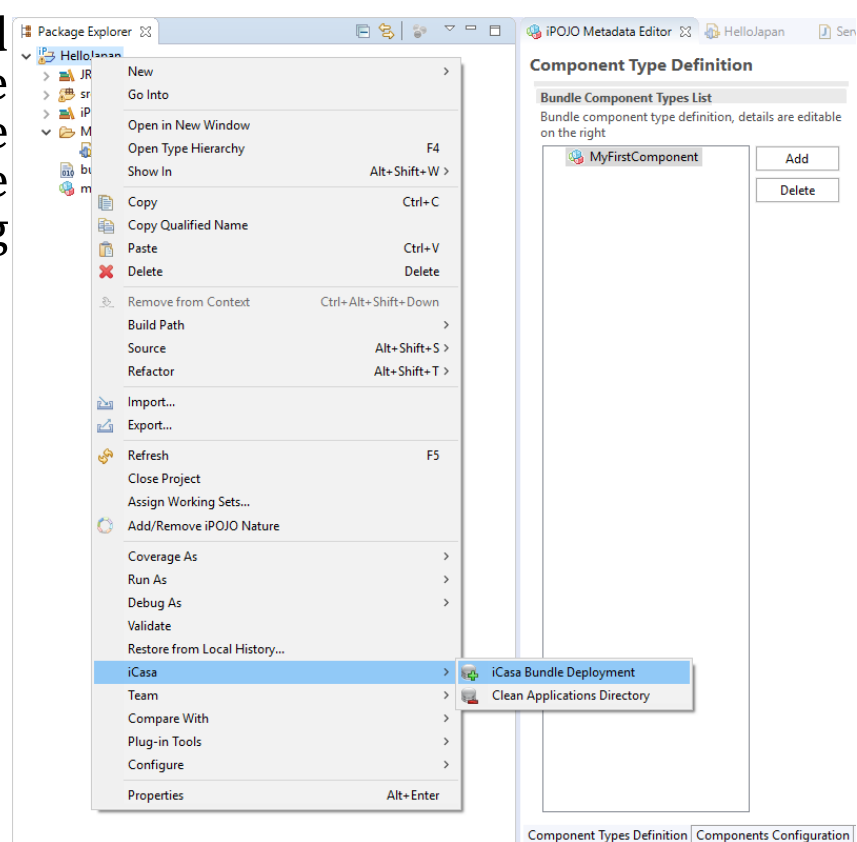


The dependency you described are wrapped as regular java field in your code. So to use them, just manipulate these fields. The name of the attribute must correspond to the id of the component dependency description section.

ECLIPSE ENVIRONMENT

Deployment feature:

- It compiles your code. Ensure that all your file are saved.
- It will copy the compiled artifact to the runtime directory. The one that you have indicated during the configuration phase of the plug in in preferences section.



RUNTIME ENVIRONMENT

A dynamic interactive runtime environment:

- Folder + Configuration files on your laptop
- Use launch script to start the runtime environment (startGateway.sh on Unix based system or startGateway.bat on windows)

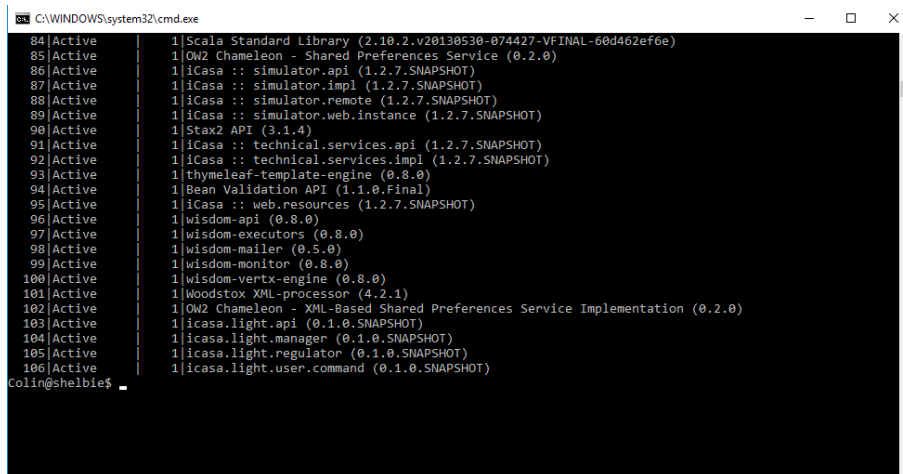


The deployment action in Eclipse Environment copy the generated bundle in this directory. If the runtime environment is started, it continuously watch this directory and integrated the new code in the execution.

RUNTIME ENVIRONMENT

A dynamic interactive runtime environment:

- An interactive shell to understand what happens at execution time.
- Useful commands:
 - *lb* : for list bundle – List of the bundle present in the environment and their state
 - *instances* : list all the instances of iPOJO component present in the environment and their states.
 - *instance \$id* : print the description of a particular instance. It shows the state of the different features of an iPOJO instance (state of the dependency, provided services,...)



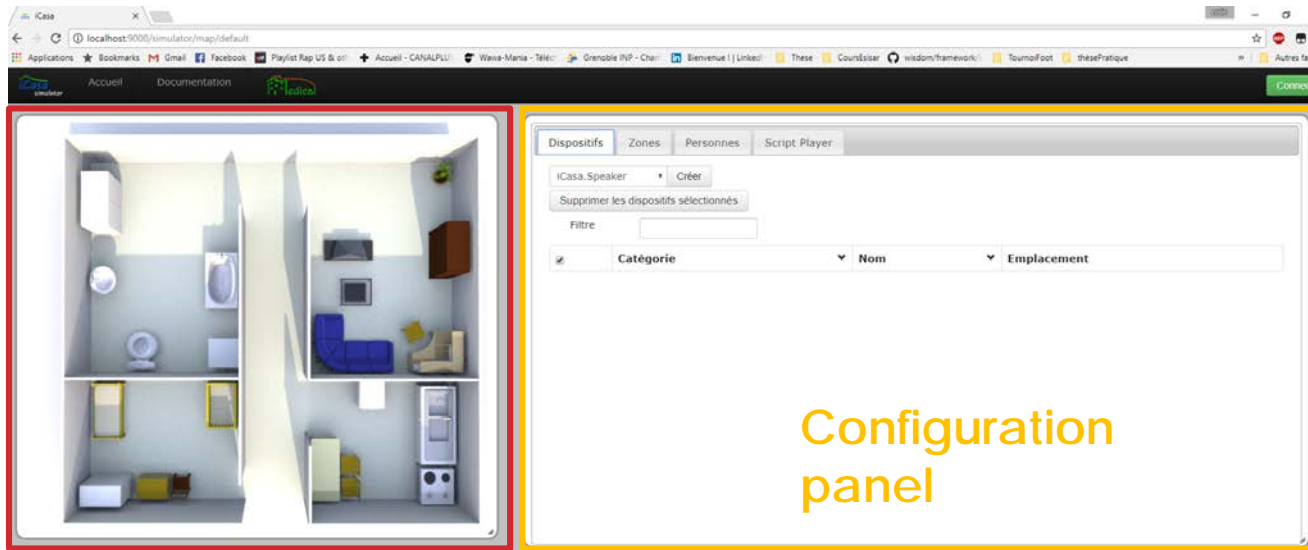
```
C:\WINDOWS\system32\cmd.exe
84|Active      | 1|Scala Standard Library (2.10.2.v20130530-074427-VFINAL-60d462ef6e)
85|Active      | 1|OW2 Chameleon - Shared Preferences Service (0.2.0)
86|Active      | 1|iCasa :: simulator.api (1.2.7.SNAPSHOT)
87|Active      | 1|iCasa :: simulator.impl (1.2.7.SNAPSHOT)
88|Active      | 1|iCasa :: simulator.remote (1.2.7.SNAPSHOT)
89|Active      | 1|iCasa :: simulator.web.instance (1.2.7.SNAPSHOT)
90|Active      | 1|Stax2 API (3.1.4)
91|Active      | 1|iCasa :: technical.services.api (1.2.7.SNAPSHOT)
92|Active      | 1|iCasa :: technical.services.impl (1.2.7.SNAPSHOT)
93|Active      | 1|thymeleaf-template-engine (0.8.0)
94|Active      | 1|Bean Validation API (1.1.0.Final)
95|Active      | 1|iCasa :: web.resources (1.2.7.SNAPSHOT)
96|Active      | 1|wisdom-api (0.8.0)
97|Active      | 1|wisdom-executors (0.8.0)
98|Active      | 1|wisdom-mailer (0.5.0)
99|Active      | 1|wisdom-monitor (0.8.0)
100|Active     | 1|wisdom-vertx-engine (0.8.0)
101|Active     | 1|Woodstox XML-processor (4.2.1)
102|Active     | 1|OW2 Chameleon - XML-Based Shared Preferences Service Implementation (0.2.0)
103|Active     | 1|icasa.light.api (0.1.0.SNAPSHOT)
104|Active     | 1|icasa.light.manager (0.1.0.SNAPSHOT)
105|Active     | 1|icasa.light.regulator (0.1.0.SNAPSHOT)
106|Active     | 1|icasa.light.user.command (0.1.0.SNAPSHOT)
colin@shelbie$
```

WEB UI

A graphical view of the execution environment:

- Available on any web browser if the Runtime Environment is started
- Address : <http://localhost:9000/simulator>

Map of
the
house



Configuration
panel

WEB UI

Creation of simulated devices :

Choose a simulated device type

Ask the execution environment to create a new simulated device of the specified type

The screenshot shows the iCasa simulator web interface. On the left is a 3D rendering of a house interior. On the right is a control panel with tabs for 'Dispositifs', 'Zones', 'Personnes', and 'Script Player'. The 'Dispositifs' tab is active, showing a dropdown menu with 'iCasa.PresenceSen' selected and a 'Créer' button. Below this is a 'Supprimer les dispositifs sélectionnés' button and a 'Filtre' input field. A table titled 'List of running devices' is displayed, containing three rows of device information.

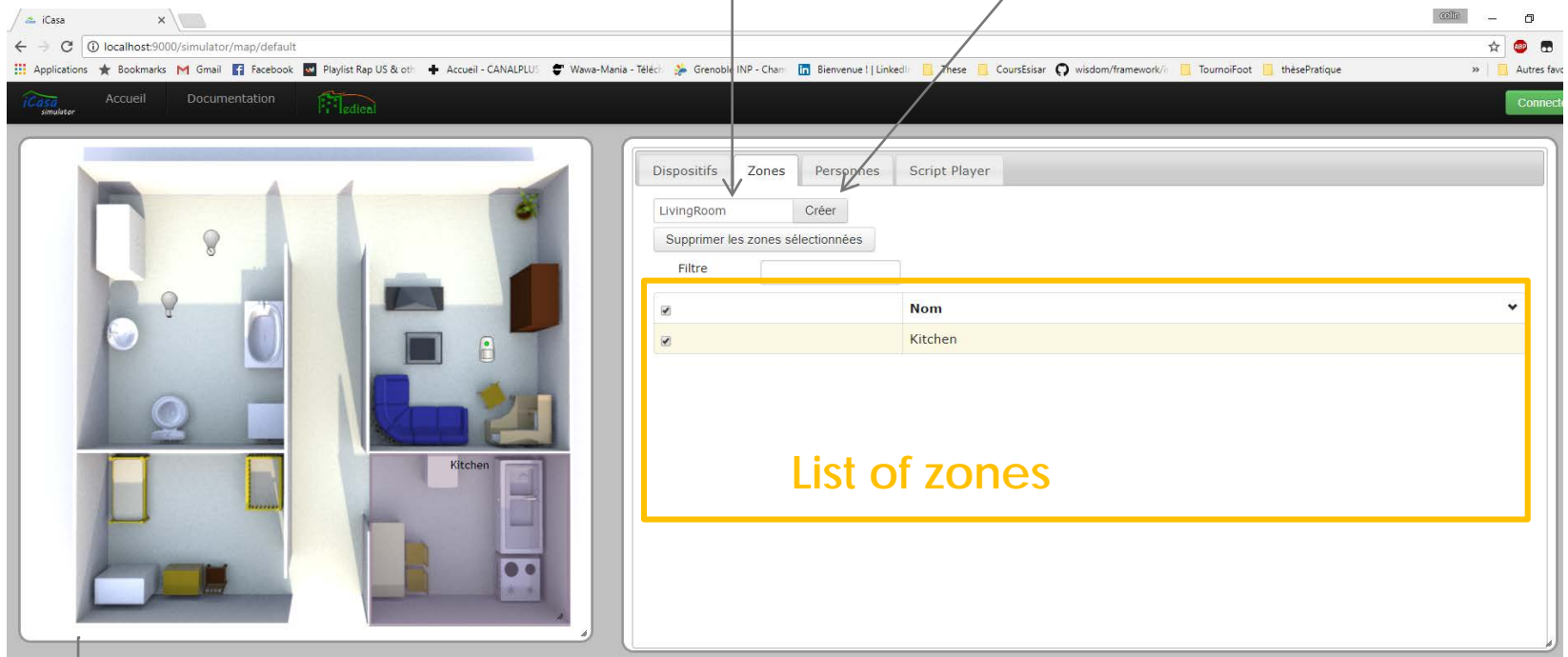
	Catégorie	Nom	Emplacement
	Light	BinaryLight-590c1d4dce	unknown
	Light	DimmerLight-aeade31aad	unknown
	Sensor	PresenceSensor-4cf64c9b29	unknown

WEB UI

Creation of simulated zones:

Provide the zone name

Ask the execution environment to create a new zone with the name you provide



Use the map to dynamically adjust the size and the position of the zone

WEB UI

Creation of simulated users:

Ask the execution environment to create a new simulated user

Provide the simulated user name

The screenshot shows the iCasa simulator web interface. On the left is a 3D map of a house with rooms labeled 'LivingRoom' and 'Kitchen'. On the right is a 'Personnes' (People) management panel. This panel includes a 'Créer' (Create) button, a 'Supprimer les personnes sélectionnées' (Delete selected people) button, and a table listing existing simulated users.

Catégorie	Nom
	Philippe
	Eva

List of simulated users

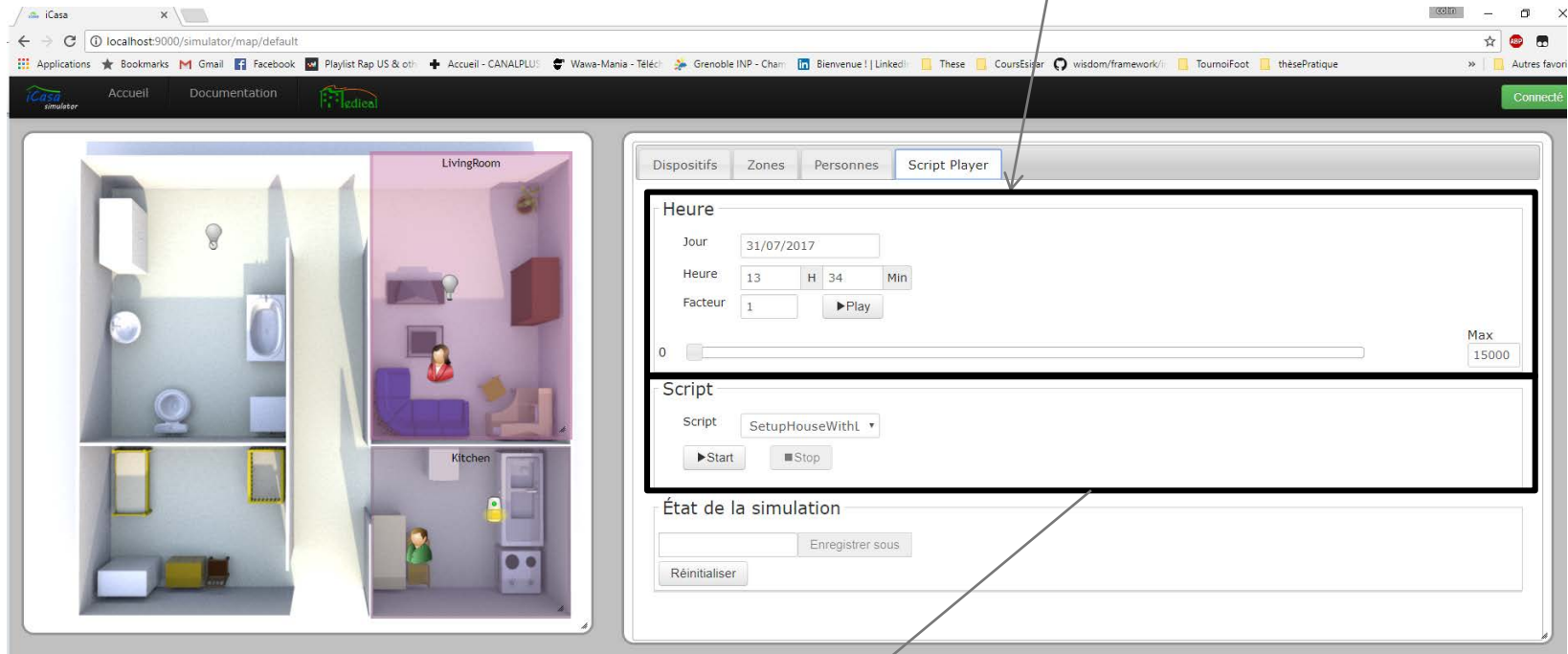
Use the map to move the user in the house

WEB UI

Script player :

Clock control panel:

- Start/Stop the simulated clock
- Several physical models of the simulator (like the temperature) need the clock to be started in order to work



All of the previous actions can be automated and contains in a script file to speed up the configuration of the execution environment

STRUCTURE

Generality about iCasa

Details

Expected exercises

EXPECTED EXERCICES

Understand bundle, component lifecycle.

- Create and deploy a bundle that contains a single component. This component have two lifecycle method that will automatically be called by the runtime each time you deployed/undeployed it.

http://self-star.imag.fr/?page_id=196

EXPECTED EXERCICES



The Light Follow Me Application :

- Try to provide a luminous path to user inside the house.

Basic behavior :

- Each time a user enter in a zone (the presence is detected by presence sensor), the application turn on the binary light of the zone. If the user leave the zone, the application turn off the light.

http://self-star.imag.fr/?page_id=61

EXPECTED EXERCICES

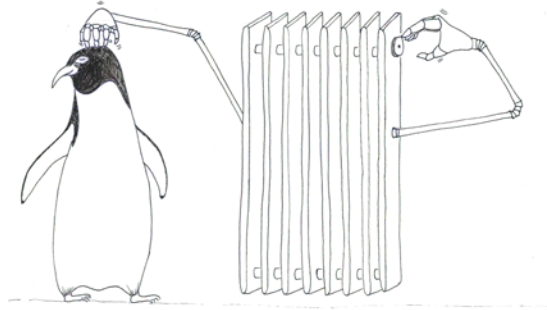


Optional Features to integrate gradually:

- Control more device type (DimmerLight)
- Manage an autonomic illuminance goal
- Manage an autonomic energy goal

<http://self-star.imag.fr/?theme=theme-1>

EXPECTED EXERCICES



The Temperature Follow Me Application :

- Try to regulate the temperature in the house.

Basic behavior :

- Try to reach a goal temperature in each room of the house through the .

<http://self-star.imag.fr/?theme=temperature-management>