**The Future Internet Public-Private Partnership (FI-PPP)**

**Goal:** capture new opportunities derived from Future Internet technology trends
- Broadband connectivity, IoT, Cloud, Big Data, etc

**Approach:** boost innovation by fostering industry-driven ecosystem
- Generic Platform (FIWARE)
- Industry-specific platforms and trials (‘Use-Cases’)  
- Broader community of developers and entrepreneurs

EC provides half of the funding:

**FACTS:**
- Industry driven, major industry players involved
- 2s ≤ 300 million
- 158 68% 18
- 23 countries represented (2 from outside Europe)

**Pan-european dimension**
FIWARE: Targeting developers needs

**What**

- Connect apps to the physical world
- Manage open data at large scale and transform it into knowledge
- Benefit from open innovation (crowd-sourcing, apps composition)
- Reach target users, monetize
- Ensuring Privacy, Security and Trust
- Take the most of infrastructures while keeping costs lower and under control
- Access from everywhere, adapt to devices

**How**

- IoT-M2M Enablers
- Data/Context Enablers
- Integration and Composition Enablers
- Business & Delivery Framework (revenue-share, cross-selling, ...)
- Security Enablers
- Advanced Cloud Enablers
- Enablers easing interface to Network and Devices

Built-in APIs & tools
FIWARE: Major Technical Chapters

**How**

1. IoT-M2M Enablers
2. Data/Context Enablers
3. Integration and Composition Enablers
4. Business & Delivery Framework (revenue-share, cross-selling, …)
5. Security Enablers
6. Advanced Cloud Enablers
7. Enablers easing interface to Network and Devices

**Technical Chapters**

1. IoT Services Enablement
2. Data/Context Management
3. Apps/Services Ecosystem & Delivery
4. Security
5. Cloud
6. I2ND

FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
FIWARE Generic Enablers (GEs)

- A FIWARE Generic Enabler (GE):
  - set of general-purpose platform functions available through APIs
  - Building with other GEs a FIWARE Reference Architecture

- FIWARE GE Specifications are open (public and royalty-free)

- FIWARE GE implementation (FIWARE GEi):
  - Platform product that implements a given GE Open Spec
  - There might be multiple compliant GEis of each GE Open Spec
  - Available FIWARE GEis published on the FIWARE Catalogue

- The FIWARE project will deliver at least one reference implementation of FIWARE GEs:
  - Based upon results of previous R&D projects
  - Publicly available Technical Roadmap updated in every release
  - Licensed with no costs within the FI-PPP program
  - Commercialized under FRAND conditions or license as open source
FIWARE Instances

- Future Internet Applications run on top of “FIWARE Instances” that are built by “FIWARE Instance Providers” upon:
  - selection of FIWARE GEis (products) from the FIWARE Catalogue
  - assembly of selected FIWARE GEis with proprietary added-value products
**FIWARE major differential features**

| Cloud          | - Federation of infrastructures (private/public regions)  
|               | - Automated GE deployment  
| Data          | - Complete Context Management Platform  
|              | - Integration of Data and Media Content  
| IoT           | - Easy plug&play of devices using multiple protocols  
|              | - Automated Measurements/Action $\leftrightarrow$ Context updates  
| Apps          | - Visualization of data (operation dashboards)  
|              | - Publication of data sets/services  
| MiWi          | - 3D and AR visualization of context  
|              | - Advanced middleware  
| Security      | - Security Monitoring  
|              | - Built-in Identity/Access/Privacy Management  
| I2ND          | - QoS across datacenters (OpenFlow)  

**FIWARE Based Application Development**  
Prof. Flávio de Oliveira Silva, Ph.D.
FIWARE Partners

- Some project partners
FIWARE Objectives (to UFU and USP)

- Create the conditions, at each region where the FIILAB is deployed, to support the development of innovative services and applications by local entrepreneurs
- Explore FIWARE Generic Enablers (GE), and Specific Enablers (SE) in order to reduce the time to market of these new services and applications
- Deploy FI-Lab Nodes in Brazil, creating the conditions for entrepreneurs to reach the market
- Collaborate with Future Internet initiatives fostering the use of technology that may help to provide a better living
Dissemination Model

- Focus on the region’s potential business models, according the local context and ecosystem
- Each FIILAB will have the participation of four stakeholders of the innovation process:
  - Higher Education Institutes (HEI)
  - Start-ups and Small and Medium Enterprises (SMEs);
  - Incubators and Accelerators
  - Funding agencies and financing entities.
- Explore opportunities regarding Smart Cities, Internet of things, Mobile devices and Applications
FIWARE Architecture

Envisioned target Smart City platform

- Smart city platform as a Data/Knowledge Hub
- Non-intrusive, open to third parties
FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
FIWARE Applications

- Reasoning

FIWARE Based Applications

- Based on challenges to developers, FIWARE fosters the creation of new services and applications that use FIWARE architecture.

- Some examples of applications that are based on FIWARE enablers:
  - [http://www.fiware.org/challenges/](http://www.fiware.org/challenges/)
  - [http://www.fiware.org/category/events/challenges-events/](http://www.fiware.org/category/events/challenges-events/)
  - [http://www.fiware.org/tag/challenges/](http://www.fiware.org/tag/challenges/)
Sample Application
FI-GUARDIAN

- Monitor for adverse events
- Created by VM9, an SME from Nova Friburgo, Brazil
- Short Presentation
  - [https://www.youtube.com/watch?v=UKfHfZRbZZA](https://www.youtube.com/watch?v=UKfHfZRbZZA)
- Winner at the FIWARE contest held at Campus Party 2014 in the Smart Cities category. Prize 75K €
FI-GUARDIAN
Building Blocks

- Based on several generic enablers
FI-GUARDIAN
Architecture Overview

FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
FIWARE Resources

- Base Site
  - http://www.fiware.org

- FIWARE Enablers Catalog
  - http://catalogue.fiware.org/enablers

- FIWARE E-Learning Platform
  - http://edu.fiware.org/

- FIWARE LAB
  - http://lab.fiware.org
Publish/Subscribe Context Broker - Orion Context Broker

- Catalog

- Documentation

- E-Learning Platform

- Long video tutorial
  - https://www.youtube.com/watch?v=tzjCA1Uhhe8

- Programmers Guide
Context management

- Context Management in FIWARE is about management of Context data (aka Context Information)
- Context Information is always relevant to “entities”, although entities can be anything (applications, users, things, …)
- Orion Context Broker intermediates between context producers and context consumers

Slides prepared by Fermín Galán Márquez (fermin@tid.es) Telefónica I+D
Orion Context Broker in a nutshell

Context Producers

- update
- update
- update

Orion Context Broker

subscriptions

1026

DB

Context Consumers

- query
- notify
- notify
NGSI API

- Based on Next Generation Services Interface (NGSI), published by Open Mobile Alliance (OMA)
- REST API (XML & JSON rendering)
- Additional "convenience" operations to ease some operations

Context Availability Management (NGSI9)
- Register context sources (registerContext)
- Search for context sources (discoverContextAvailability)
- Subscribe to context sources availability notifications (subscribeContextAvailability, updateContextAvailabilitySubscription & unsubscribeContextAvailability)

Context Management (NGSI10)
- Update context information (updateContext)
- Query context information (queryContext)
- Subscribe to context information notifications (subscribeContext, updateContextSubscription & unsubscribeContext)

23
The NGSI information model

- Context Element
  - EntityId
  - EntityType

- Context Element attributes
  - Name
  - Type
  - Value

- Meta-data
  - Name
  - Type
  - Value

Orion metadata: ID, location
Custom metadata

Node Lamps
Electrical Regulator
Vans
Technicians

Issues

Severity
Battery Charge
Electric Potential
Presence
Location
Time Instant
Illuminance
Entity Creation (Request)

- Using an updateContext

**XML**
```
(curl localhost:1026/NOSI10/updateContext -s -S --header 'Content-Type: application/xml' -d @- | xmlint --format -) <<EOF
<?xml version="1.0" encoding="UTF-8"?>
<updateContextRequest>
  <contextElementList>
    <contextElement>
      <entityId type="Room" isPattern="false">
        <id>Room1</id>
      </entityId>
      <contextAttributeList>
        <contextAttribute>
          <name>temperature</name>
          <type>centigrade</type>
          <contextValue>23</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>pressure</name>
          <type>mmHg</type>
          <contextValue>720</contextValue>
        </contextAttribute>
      </contextAttributeList>
    </contextElement>
  </contextElementList>
  <updateAction>APPEND</updateAction>
</updateContextRequest>
EOF
```

**JSON (since release 0.9.0)**
```
(curl localhost:1026/NOSI10/updateContext -s -S --header 'Content-Type: application/json' -d @- | python -mjson.tool) <<EOF
{
  "contextElements": [
    {
      "type": "Room",
      "isPattern": "false",
      "id": "Room1",
      "attributes": [
        {
          "name": "temperature",
          "type": "centigrade",
          "value": "23"
        },
        {
          "name": "pressure",
          "type": "mmHg",
          "value": "720"
        }
      ]
    }
  ],
  "updateAction": "APPEND"
}
EOF
```
Entity Creation (Response)

- Using an `updateContext`

**XML**

```xml
<?xml version="1.0"?>
<updateContextResponse>
  <contextResponseList>
    <contextElementResponse>
      <contextElement>
        <entityId type="Room" isPattern="false">
          <id>Room1</id>
        </entityId>
        <contextAttributeList>
          <contextAttribute>
            <name>temperature</name>
            <type>centigrade</type>
            <contextValue/>
          </contextAttribute>
          <contextAttribute>
            <name>pressure</name>
            <type>mmHg</type>
            <contextValue/>
          </contextAttribute>
        </contextAttributeList>
        <contextAttribute>
          <name>temperature</name>
          <type>centigrade</type>
          <contextValue/>
        </contextAttribute>
        <contextAttribute>
          <name>pressure</name>
          <type>mmHg</type>
          <contextValue/>
        </contextAttribute>
      </contextElement>
    </contextElementResponse>
    <contextElementResponse>
      <contextElement>
        <entityId type="Room" isPattern="false">
          <id>Room1</id>
        </entityId>
        <contextAttributeList>
          <contextAttribute>
            <name>temperature</name>
            <type>centigrade</type>
            <contextValue/>
          </contextAttribute>
          <contextAttribute>
            <name>pressure</name>
            <type>mmHg</type>
            <contextValue/>
          </contextAttribute>
        </contextAttributeList>
        <contextAttribute>
          <name>temperature</name>
          <type>centigrade</type>
          <contextValue/>
        </contextAttribute>
        <contextAttribute>
          <name>pressure</name>
          <type>mmHg</type>
          <contextValue/>
        </contextAttribute>
      </contextElement>
    </contextElementResponse>
  </contextResponseList>
  <statusCode>
    <code>200</code>
    <reasonPhrase>OK</reasonPhrase>
  </statusCode>
</updateContextResponse>
```

**JSON (since release 0.9.0)**

```json
{  "contextResponses": [  {    "contextElement": {      "attributes": [        {          "name": "temperature",          "type": "centigrade",          "value": ""        },        {          "name": "pressure",          "type": "mmHg",          "value": ""        }      ]    },    "id": "Room1",    "isPattern": "false",    "type": "Room"  }  ],  "statusCode": {    "code": "200",    "reasonPhrase": "OK"  } }
```
Query Context (request)

```xml
(curl localhost:1026/NGSI10/queryContext -s -S --header 'Content-Type: application/xml' -d @- |
xml -l int --format -) <<<EOF
<?xml version="1.0" encoding="UTF-8"?>
<queryContextRequest>
  <entityIdList>
    <entityId type="Room" isPattern="false">
      <id>Room1</id>
    </entityId>
  </entityIdList>
</queryContextRequest>
EOF
```

```json
(curl localhost:1026/NGSI10/queryContext -s -S --header 'Content-Type: application/json' --header 'Accept: application/json' -d @- | python -m json.tool) <<<EOF
{
  "entities": [
    {
      "type": "Room",
      "isPattern": "false",
      "id": "Room1"
    }
  ]
}
EOF
```
XML

```xml
<queryContextResponse>
  <contextResponseList>
    <contextElementResponse>
      <contextElement>
        <entityId type="Room" isPattern="false">
          <id>Room1</id>
        </entityId>
        <contextAttributeList>
          <contextAttribute>
            <name>temperature</name>
            <type>centigrade</type>
            <contextValue>23</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>pressure</name>
            <type>mmHg</type>
            <contextValue>720</contextValue>
          </contextAttribute>
        </contextAttributeList>
      </contextElement>
    </contextElementResponse>
  </contextResponseList>
  <statusCode>
    <code>200</code>
    <reasonPhrase>OK</reasonPhrase>
  </statusCode>
</queryContextResponse>
```

JSON (since release 0.9.0)

```json
{
  "contextResponses": [
    {
      "contextElement": {
        "attributes": [
          {
            "name": "temperature",
            "type": "centigrade",
            "value": "23"
          },
          {
            "name": "pressure",
            "type": "mmHg",
            "value": "720"
          }
        ],
        "id": "Room1",
        "isPattern": "false",
        "type": "Room"
      },
      "statusCode": {
        "code": "200",
        "reasonPhrase": "OK"
      }
    }
  ]
}
```
FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.

FIWARE Context/Data Management Platform

OMA NGSI-9/10

Distributed Context Sources

Programming of rules

Applications

Gathered data injected for CEP-like processing

Data generated either by CEP or BigData is published

Gathered data is injected for processing/analysis

Processed data is injected for processing/analysis

Complex Event Processing (PROTON)

BigData (COSMOS)

Context/Data Management Platform

Direct bigdata injection

FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
How Orion fits in the FIWARE overall platform

FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
Orion in LiveDemo application

Object Storage → Wirecloud (javascript runtime) → Wirecloud (server) → Context Broker → Ticket Management → Location

Browser widgets → Context Broker → Cosmos → CEP

History → NGSI IoT Adapter → IoT Backend Device Management

FI-Lab

FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
Application Mashup - Wirecloud

- End-user UI development by using Widget and Mashups

  - Widget
    - A small application or piece of dynamic content that can be easily placed into a web page
    - Often encapsulate a Web API (directly or through an operator)
    - Can be easily embedded into webpages (HTML snippets)
    - "Mashable" widgets generate/consume events, so that they can be wired together to create a lightweight application mashup
    - This requires a widget platform

  - Mashup
    - Lightweight application combining data, services and UIs from multiple sources
    - Developed by either IT or business staff, as well as by end users
    - Created in hours or days, not months
    - Uses a Web Oriented Architecture (WOA)
    - Often relies on internal + external web services (Web APIs)
    - Done at data, logic and/or presentation layers
Widgets and Mashups

- Widgets

- Mashups
Mashups and Operators

- Operators represent functionality managed at the presentation layer.
- Empower the user to create data mashups (Access + Transform data sources) by piping + Web API wrapping.
- Enhance widget functionality by wiring them with each other.
Application Mashup – Wirecloud

Resources

- Catalog
  - http://catalogue.fiware.org/enablers/application-mashup-wirecloud

- Documentation

- E-Learning Platform

- Short video tutorial
  - https://www.youtube.com/watch?v=yzQgstBAUeo
Setting up your virtual infrastructure using FI-LAB Cloud

- Introduction to FI-LAB Cloud Hosting
- Deploying your first VM
- Deploying components for your application
- Object Storage API
- Reference Information
FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.

FI-LAB Cloud Hosting

Your Applications

OpenStack Dashboard

APIs

OPENSTACK
CLOUD OPERATING SYSTEM

Compute
Networking
Storage

OpenStack Shared Services

Standard Hardware
FI-LAB Cloud Hosting

- Create your account in lab.fiware.eu
- Enter in the Cloud Portal
- Create your keypair (private/private key)
- Deploy your instance
- Add a public IP
- Open ports to the VM
Enter your email and password to access to the FI-LAB.

If you do not have it or forgot it, sign up or request for a new one.
You must create a keypair to access to the servers.
Choose the VM and click **Launch**.
Open ports in your VM.
Deploying components for your application

- Deploying applications and not only Servers.
  - Ad hoc installation (not template usage).
- Managing applications in Servers (install, uninstall, configure, snapshot).
- Deploying different environments for that applications.
Deploy example

Blueprint template: fiware1

- **Servers:**
  - Tier 1: Tomcat
  - Tier 2: Context Broker
  - Tier 3: MongoDB

- **Servers:**
  - 2-5 servers

- **Server:**
  - 1 server

- **Tier:** Each kind of software and server to be deployed.

- **Each Tier can be deployed in one or several servers** (e.g. tomcat, 2-5 servers).

- **Blueprint Instance:** Deployed in the testbed.

- **Blueprint Template:** platform specification to be deployed.
You can open the Catalog of Blueprint Template or create one from the scratch.

Press the option “Blueprint Templates” from menu.
You should introduce the “Name” and the “Description”.

Demo
To add new Tier, press “DemoSantander” name.
After press “Add Tier” you see this windows to define the servers of this tier.

You must define a Keypair to access to those servers.

Install software pressing the mouse right click.
You should specify the maximum, minimum and current number of servers.
Press “Action” and select “Launch Template” to launch the instance.
You should specify the “Name” and “Description” for your blueprint.
Firstly, we need to instantiate The servers.
Secondly, the installation of the software.
Finally, if all was ok.

Pressing the name you can see the tiers of this blueprint.
Press it to get information of your server.
### Demo

#### Instances

<table>
<thead>
<tr>
<th>Info</th>
<th>Specs</th>
<th>IP Addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: DemoSantanderInstance-TierApache-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID: 2a55a0c50c-05a-488b-b98e-920eb3812b6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status: ACTIVE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM: 2048MB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VCPUs: 1VCPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk: 10GB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.0.0.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>156.206.63.27</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Security Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>sg_DemoSantander_00000000000000000000000000000000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Meta</th>
<th>Volumes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key name: keypair1</td>
<td></td>
</tr>
<tr>
<td>Image Name: chef aware</td>
<td></td>
</tr>
<tr>
<td>No volumes attached.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Installed Software</th>
<th>Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>tomcat 6</td>
<td>INSTALLED</td>
</tr>
</tbody>
</table>
If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.html
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

**NOTE:** For security reasons, using the manager webapp is restricted to users with role "manager". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation, and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- users@tomcat.apache.org for general questions related to configuring and using Tomcat
- dev@tomcat.apache.org for developers working on Tomcat

Thanks for using Tomcat!
FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
Big Data Analysis - Cosmos

- Catalog

- Documentation

- E-Learning Platform

- Long video tutorial
  - [https://www.youtube.com/watch?v=JJ6QCojnCh4](https://www.youtube.com/watch?v=JJ6QCojnCh4)

- Programmers Guide

- Quick Start
Complex Event Processing (CEP)

Stream and Multimedia - Kurento


- [https://www.youtube.com/watch?v=TC5bAIVyE-M](https://www.youtube.com/watch?v=TC5bAIVyE-M)
- [https://www.youtube.com/watch?v=TBkrI3fmHWI](https://www.youtube.com/watch?v=TBkrI3fmHWI)
- [https://www.youtube.com/watch?v=5eJRnwKxgbY](https://www.youtube.com/watch?v=5eJRnwKxgbY)
- [https://www.youtube.com/watch?v=PYCw9-4oWWA](https://www.youtube.com/watch?v=PYCw9-4oWWA)
CKAN

- CKAN is an open data management platform
- helps publishers make their data accessible and findable by potential users.
  - Publisher: Data + Metadata
  - Consumer: Browse, Search, Access
  - http://ckan.org/
- CKAN Tour
  - http://ckan.org/tour/
- CKAN Features Overview
  - http://ckan.org/features/
- FIWARE
- Cygnus connector
  - https://github.com/telefonicaid/fiware-connectors/tree/master/flume
CKAN

- WireCloud Integration
  - https://github.com/wirecloud-fiware/ckan-source-operator

- Cygnus Connector
  - https://github.com/telefonicaid/fiware-connectors/tree/develop/flume
CKAN Architecture

- Application Layer
- Data Layer
Security

- OAuth 2.0
  - http://oauth.net/2/

- Course (OAuth in FI-LAB)

- OAuth Sample
  - https://github.com/ging/oauth2-example-client
Prova de Conceito MyBUS

- Aplicação baseada em Open Data (SETTRAN)
- Utiliza os Generic Enablers da Arquitetura FIWARE
- Baseada no conceito de Smart City com foco mobilidade urbana

PUBLIC DATA BUSES INFORMATION

FIWARE Based Application Development
Prof. Flávio de Oliveira Silva, Ph.D.
Prova de Conceito
MyBUS
PUBLIC DATA BUSES
INFORMATION - Generator

- Open data from the municipality of Uberlandia
- For the workshop we will use sample data
- Data Generation
  - com.br.control.Linha131
- Rest Web service
  - com.br.view.Get131
  - @Path("/get/131")
  - @GET
  - @Produce(MediaType.APPLICATION_JSON)
Integration Controller

- Standard Java console base application
  - Main
  - Calls the generator WebService
  - Post data to Orion

- Get data from the Generator Service and Post Data to Orion
  - GetInfoWebGen. GetBus
  - GET - public void getBus()
    - http://localhost:8080/WebGen/get/131
  - POST - public void sendToOrion()
Configure Orion to Notify MyBusFrontController

```
{
  "entities": [
    {
      "type": "BusCar",
      "isPattern": "false",
      "id": "1"
    }
  ],
  "attributes": ["Latitude","Longitude"],
  "duration": "P1M",
  "notifyConditions": [
    {
      "type": "ONCHANGE",
      "condValues": ["Latitude","Longitude"
    }
  ],
  "throttling": "PT1S"
}
```
Configure Orion to Notify Cygnus

```json
{
    "entities": [
        {
            "type": "BusCar",
            "isPattern": "false",
            "id": "1"
        }
    ],
    "attributes": ["Latitude","Longitude"],
    "reference": "http://localhost:5050/notify",
    "duration": "P1M",
    "notifyConditions": [
        {
            "type": "ONCHANGE",
            "condValues": ["Latitude","Longitude"
        }
    ],
    "throttling": "PT1S"
}
```
MyBusFrontController

- Receive Orion Notification
- Offers an interface for the Mobile App to get new information from a bus
- `br.com.webbus.controller.OrionController`
  - Receives a notification from Orion
  - `@RequestMapping(value="/receive", method=RequestMethod.POST)`
  - `public @ResponseBody String getBusOrion`
  - Deserialize to a Java Data object
  - Stores new Data object in memory
- `br.com.webbus.controller.BusRouteController`
  - Offers buses position to the Application
  - `@RequestMapping(value="/listbus/{valor}" , method=RequestMethod.GET)`
  - `public @ResponseBody List<Bus> getListBus`
Bus Mobile App

- Activity responsible to get buses positions
- br.com.busapp.FindBusactivity