V2_ Knowledgebase Visualization Interface

Carles Tardío Pi, report made in LaTeX

July, 2012

Abstract

This document is a technical report of the knowledgebase visualization interface. Here I will describe the steps followed during the devolopment of the project. For any question regarding the knowledgebase graph visualizations you can contact me at carlestapi@gmail.com.

1 Knowledgebase

The `knowledgebase.projects.nl` web page contains 3 knowledgebases about current research topics at V2 lab. The wearable technology knowledgebase that was created by Silvia Piantini and is maintained by Piem Wirtz, the augmented reality that was created by Carolien Teunisse and is maintained by Jan Misker, and the ecology that was created by Carles Tardio Pi and is maintained by Michel van Dartel.
All the project information data can be uploaded, edited and published by the mainters of the knowledgebases with the content management framework Joomla. All that information is stored in v2 Macula server and its access goes through the `https://mysql.v2.nl` MYSQL database management system. The way that knowedgebase information is structured within MYSQL is through 6 databases.

- Wearable Technology Knowledgebase
  - **j25_knowledgebase_wearable**
    1. *name:* id ; *type:* int(10)
    2. *name:* projectname ; *type:* varchar(255) utf8_general_ci
    3. *name:* organisation ; *type:* varchar(255) utf8_general_ci
    4. *name:* persons ; *type:* varchar(255) utf8_general_ci
    5. *name:* city ; *type:* varchar(100) utf8_general_ci
    6. *name:* country ; *type:* varchar(100) utf8_general_ci
    7. *name:* text ; *type:* utf8_general_ci
    8. *name:* url ; *type:* varchar(255) utf8_general_ci
    9. *name:* tags ; *type:* utf8_general_ci
    10. *name:* period ; *type:* varchar(20) utf8_general_ci
  - **j25_knowledgebase_wearable_tags**
    1. *name:* id ; *type:* int(11)
    2. *name:* itemid ; *type:* int(11)
    3. *name:* tag ; *type:* varchar(100) utf8_general_ci
- Augmented Reality Knowledgebase
  - **j25_knowledgebase_ar**
    1. *name:* id ; *type:* int(10)
    2. *name:* projectname ; *type:* varchar(255) utf8_general_ci
    3. *name:* organisation ; *type:* varchar(255) utf8_general_ci
    4. *name:* persons ; *type:* varchar(255) utf8_general_ci
    5. *name:* city ; *type:* varchar(100) utf8_general_ci
    6. *name:* country ; *type:* varchar(100) utf8_general_ci
    7. *name:* text ; *type:* utf8_general_ci
    8. *name:* url ; *type:* varchar(255) utf8_general_ci
    9. *name:* tags ; *type:* utf8_general_ci

10. *name:* period ; *type:* varchar(20) utf8_general_ci

- **j25_knowledgebase_ar_tags**
  1. *name:* id ; *type:* int(11)
  2. *name:* itemid ; *type:* int(11)
  3. *name:* tag ; *type:* varchar(100) utf8_general_ci

- Ecology Knowledgebase
  - **j25_knowledgebase_ecology**
    1. *name:* id ; *type:* int(10)
    2. *name:* projectname ; *type:* varchar(255) utf8_general_ci
    3. *name:* organisation ; *type:* varchar(255) utf8_general_ci
    4. *name:* persons ; *type:* varchar(255) utf8_general_ci
    5. *name:* city ; *type:* varchar(100) utf8_general_ci
    6. *name:* country ; *type:* varchar(100) utf8_general_ci
    7. *name:* text ; *type:* utf8_general_ci
    8. *name:* url ; *type:* varchar(255) utf8_general_ci
    9. *name:* tags ; *type:* utf8_general_ci
    10. *name:* period ; *type:* varchar(20) utf8_general_ci
  - **j25_knowledgebase_ecology_tags**
    1. *name:* id ; *type:* int(11)
    2. *name:* itemid ; *type:* int(11)
    3. *name:* tag ; *type:* varchar(100) utf8_general_ci

2 Gephi & Gephi Toolkit

Gephi is an open-source software for visualizing and analysing large networks graphs. Gephi uses a 3D render engine to display graphs in real-time and speed up the exploration. You can use it to explore, analyse, spatialise, filter, clusterize, manipulate and export all types of graphs.
Because the necessity of having an update response between the the knowledgebase and the interface we need to be able to run the program within a Java command-line terminal in the server. This is done through the Gephi Toolkit.
The Gephi Toolkit project package essential modules in a standard Java library, which any Java project can use for getting things done. The toolkit is just a single JAR that anyone could reuse in new Java applications and achieve tasks that can be done in Gephi automatically, from a command-line program for instance.
Gephi is designed in a modular way and splitted into different modules. All features are wrapped into separated modules, for instance a module for the graph structure, a module for the layout algorithms and so on. Moreover business modules are separated from user interfaces modules. That allows to keep only business modules and remove UI without any problems. That is the purpose of the toolkit, which wraps only core modules and remove all the UI layer.

Figure 1: Gephi architecture and modules included in the Toolkit

2.1 Built Gephi Toolkit in Netbeans Platform

The first step that has to be done is build the latest version of Gephi Toolkit from the source code. Gephi source code is hosted on Github platform, and use Git for source control. You can get latest source revisions by following instructions here: `https://wiki.gephi.org/index.php/Checkout_Code`. Our folder name that contains it is: gephi-0.8.1-beta.sources.

Because Gephi is built on top of the Netbeans Platform, all development needs to take place in Netbeans IDE, which fully integrates all tools for Netbeans Platform developers.
The NetBeans Platform is a generic framework for Swing applications. It provides the "plumbing" that, before, every developer had to write themselves—saving state, connecting actions to menu items, toolbar items and keyboard shortcuts; window management, and so on.
Gephi software architecture is modular and therefore each feature is split into modules. Modules depend on each other, similar to Java packages. Plugins developers simply create new modules that contains their code, add dependencies to Gephi modules, and distribute their plugins by creating an NBM package.

Figure 2: Gephi Architecture

You can check the steps how to build Gephi in Netbeans or from the command-line here: `https://wiki.gephi.org/index.php/Configuring_NetBeans`.

2.2 Program Coding

Here are explained in detail the different Toolkit modules by following the Gephi hierarchy:

2.2.1 First Steps - Import Packages and Plugins

```
package org.gephi.toolkit.demos;

import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import org.gephi.data.attributes.api.AttributeController;
import org.gephi.data.attributes.api.AttributeModel;
import org.gephi.graph.api.GraphController;
import org.gephi.graph.api.GraphModel;
import org.gephi.graph.api.Node;
import org.gephi.graph.api.UndirectedGraph;
import org.gephi.io.database.drivers.MySQLDriver;
import org.gephi.io.database.drivers.SQLUtils;
import org.gephi.io.importer.api.Container;
import org.gephi.io.importer.api.EdgeDefault;
import org.gephi.io.importer.api.ImportController;
import org.gephi.io.importer.plugin.database.EdgeListDatabaseImpl;
importorg.gephi.io.importer.plugin.database.ImporterEdgeList;
import org.gephi.io.processor.plugin.DefaultProcessor;
import org.gephi.layout.plugin.force.StepDisplacement;
import org.gephi.layout.plugin.force.yifanHu.YifanHuLayout;
import org.gephi.project.api.ProjectController;
import org.gephi.project.api.Workspace;
import org.openide.util.Lookup;

import java.awt.Color;
import java.io.IOException;
import org.gephi.data.attributes.api.AttributeColumn;
import org.gephi.data.attributes.api.AttributeController;
```

```
importorg.gephi.data.attributes.api.AttributeModel;
import org.gephi.filters.api.FilterController;
import org.gephi.filters.api.Query;
importorg.gephi.filters.api.Range;
import org.gephi.filters.plugin.graph.DegreeRangeBuilder.DegreeRangeFilter;
import org.gephi.graph.api.DirectedGraph;
import org.gephi.graph.api.GraphController;
import org.gephi.graph.api.GraphModel;
import org.gephi.graph.api.GraphView;
import org.gephi.graph.api.UndirectedGraph;
import org.gephi.io.exporter.api.ExportController;
import org.gephi.io.importer.api.Container;
import org.gephi.io.importer.api.EdgeDefault;
import org.gephi.io.importer.api.ImportController;
import org.gephi.io.processor.plugin.DefaultProcessor;
import org.gephi.layout.plugin.force.StepDisplacement;
import org.gephi.layout.plugin.force.yifanHu.YifanHuLayout;
import org.gephi.layout.plugin.AbstractLayout;
import org.gephi.preview.api.PreviewController;
import org.gephi.preview.api.PreviewModel;
import org.gephi.preview.api.PreviewProperty;
import org.gephi.preview.types.EdgeColor;
import org.gephi.project.api.ProjectController;
import org.gephi.project.api.Workspace;
import org.gephi.ranking.api.Ranking;
import org.gephi.ranking.api.RankingController;
import org.gephi.ranking.api.Transformer;
import org.gephi.layout.api.LayoutController;
import org.gephi.ranking.plugin.transformer.AbstractColorTransformer;
import org.gephi.ranking.plugin.transformer.AbstractSizeTransformer;
import org.gephi.statistics.plugin.GraphDistance;
import org.openide.util.Lookup;
```

2.2.2 Initialize a project - and therefore a workspace

```
ProjectController pc = Lookup.getDefault().lookup(ProjectController.class);
pc.newProject();
Workspace workspace = pc.getCurrentWorkspace();
```

2.2.3 Get models and controllers for this new workspace - that they will be useful later

```
AttributeModel attributeModel = Lookup.getDefault().lookup(AttributeController.class).getModel();
GraphModel graphModel = Lookup.getDefault().lookup(GraphController.class).getModel();
PreviewModel model = Lookup.getDefault().lookup(PreviewController.class).getModel();
ImportController importController = Lookup.getDefault().lookup(ImportController.class);
FilterController filterController = Lookup.getDefault().lookup(FilterController.class);
RankingController rankingController = Lookup.getDefault().lookup(RankingController.class);
```

2.2.4 Import Database

To import data from the MySQL database. The database format must be "Edge List", basically a table for nodes and a table for edges. To be found by the importer, you need to have following columns:

- **Nodes**: ID and LABEL
- **Edges**: SOURCE, TARGET and WEIGHT

Any other column will be imported as attributes. Other recognized columns are X, Y and SIZE for nodes and ID and LABEL for edges.

```
EdgeListDatabaseImpl db = new EdgeListDatabaseImpl();
db.setDBName("knowledgebase");
db.setHost("mysql.v2.nl");
db.setUsername("kbread");
db.setPasswd("aeng8aXu");
db.setSQLDriver(new MySQLDriver());
//db.setSQLDriver(new PostgreSQLDriver());
//db.setSQLDriver(new SQLServerDriver());
db.setPort(3306);
db.setNodeQuery("SELECT DISTINCT tag AS id, tag AS label, 'project tag' FROM `j25_knowledgebase
_ecology_tags` UNION SELECT id, projectname AS label , url FROM `j25_knowledgebase_ecology`");
db.setEdgeQuery("SELECT tag AS source, item_id AS target FROM `j25_knowledgebase_ecology_tags`");
ImporterEdgeList edgeListImporter = new ImporterEdgeList();
```

2.2.5 Don't create missing nodes

```
container.setAllowAutoNode(false);
```

2.2.6 Force Undirected

```
container.getLoader().setEdgeDefault(EdgeDefault.UNDIRECTED);
```