

# RefactoredGraphVisualizerInlineTemps.java

```

1 package lehmanrulesgraphsdisplay;
2
3 import java.util.ArrayList;
17
18 public class RefactoredGraphVisualizerInlineTemps {
19     private History h;
20
21     public RefactoredGraphVisualizerInlineTemps(History h){
22         this.h = h;
23     }
24
25     public ChartPanel []firstRuleGraphGenerator(){
26         ChartPanel []panels= new ChartPanel[2];
27
28         DefaultCategoryDataset objDataset = createChangesDataset();
29         panels[0] = createBarChartPanel("Version per Year Bar Chart", "Version ID",
"changes", objDataset);
30
31         DefaultCategoryDataset objDataset2 = createVersionsDataset();
32         panels[1] = createBarChartPanel("Version per Year Bar Chart", "Version ID",
"Versions", objDataset2);
33
34
35         return panels;
36     }
37 }
38
39 // TODO move to History ?
40 private DefaultCategoryDataset createChangesDataset() {
41     DefaultCategoryDataset objDataset = new DefaultCategoryDataset();
42
43     for(int i=0; i<h.getVersions().size(); i++){
44         objDataset.setValue(getOperationChangesPerVersion(i),
45             "Operations", getXAxisLabelPerVersion(i));
46         objDataset.setValue(getStructsChangesPerVersion(i),
47             "Structs", getXAxisLabelPerVersion(i));
48     }
49     return objDataset;
50 }
51
52 // TODO move to History ?
53 private double getOperationChangesPerVersion(int i) {
54     return (double) h.getVersions().get(i).getopsChanges().getNumOfChanges();
55 }
56
57 // TODO move to History ?
58 private double getStructsChangesPerVersion(int i) {
59     return (double) h.getVersions().get(i).getstructsChanges().getNumOfChanges();
60 }
61
62 // TODO move to History ?
63 private String getXAxisLabelPerVersion(int i) {
64     return h.getVersions().get(i).getID();
65 }
66
67 // TODO move to History ?
68 private DefaultCategoryDataset createVersionsDataset() {
69
70     if(h.getVersions().size()<=0){
71         return null;
72     }
73

```

```

74     int [][] versionsPerYearDistribution = initializeNumOfVersionsPerYear();
75     populateNumOfVersionsPerYear(versionsPerYearDistribution);
76
77     DefaultCategoryDataset objDataset2 =
78     createVersionsPerYearDataset(versionsPerYearDistribution);
79
80     return objDataset2;
81 }
82 // TODO move to History ?
83 private int [][] initializeNumOfVersionsPerYear() {
84     int [][] versionsPerYearDistribution = new int[getProjectLifetimeInYears()][2];
85
86     for(int i=0; i<getProjectLifetimeInYears(); i++){
87         // 1st column holds a specific year label
88         versionsPerYearDistribution[i][0] = getYearOfFirstVersion() + i;
89         // 2nd column holds no of versions this year
90         versionsPerYearDistribution[i][1] = 0;
91     }
92     return versionsPerYearDistribution;
93 }
94
95 // TODO move to History ?
96 private int getProjectLifetimeInYears() {
97     return getYearOfLastVersion()-getYearOfFirstVersion()+1;
98 }
99
100 // TODO move to History ?
101 private int getYearOfLastVersion() {
102     return h.getVersions().get(h.getVersions().size()-1).yearOfDate();
103 }
104
105 // TODO move to History ?
106 private int getYearOfFirstVersion() {
107     return h.getVersions().get(0).yearOfDate();
108 }
109
110 // TODO move to History ?
111 private void populateNumOfVersionsPerYear(int[][] versionsPerYearDistribution) {
112     for(int i=0; i<h.getVersions().size(); i++){
113         for(int j=0; j<getProjectLifetimeInYears(); j++){
114             if(versionsPerYearDistribution[j][0]==getYearPerVersion(i)){
115                 versionsPerYearDistribution[j][1]++;
116                 break;
117             }
118         }
119     }
120 }
121
122 // TODO move to History ?
123 private int getYearPerVersion(int i) {
124     return h.getVersions().get(i).yearOfDate();
125 }
126
127 // TODO move to History ?
128 private DefaultCategoryDataset createVersionsPerYearDataset(int[][]
versionsPerYearDistribution) {
129     DefaultCategoryDataset objDataset2 = new DefaultCategoryDataset();
130     for(int i=0; i<getProjectLifetimeInYears(); i++){
131         String xaxis = ""+(versionsPerYearDistribution[i][0]-2000);
132         objDataset2.setValue(versionsPerYearDistribution[i][1], "Year Versions", xaxis);
133     }

```

```

134     }
135     return objDataset2;
136 }
137
138
139 private ChartPanel createBarChartPanel(String title, String xAxisTitle, String
yAxisTitle, DefaultCategoryDataset objDataset2){
140     JFreeChart objChart2 = ChartFactory.createBarChart(
141         title,
142         xAxisTitle,
143         yAxisTitle,
144         objDataset2,
145         PlotOrientation.VERTICAL,
146         true,
147         true,
148         false
149     );
150     return new ChartPanel(objChart2);
151 }
152
153 public ChartPanel[] secondRuleGraphGenerator()
154 {
155     ChartPanel []panels= new ChartPanel[2];
156     DefaultCategoryDataset objDataset = new DefaultCategoryDataset();
157     ArrayList<VersionInfo> versionlist = h.getVersions();
158     int i;
159     for(i=0; i<versionlist.size(); i++){
160         double dops = versionlist.get(i).getopsMetrics().getComplexity();
161         String xaxis = versionlist.get(i).getID();
162         objDataset.setValue(dops,"Operations", xaxis);
163     }
164     for(i=0; i<versionlist.size(); i++){
165         double dstructs = versionlist.get(i).getstructsMetrics().getComplexity();
166         String xaxis = versionlist.get(i).getID();
167         objDataset.setValue(dstructs,"Structs", xaxis);
168     }
169     JFreeChart objChart = ChartFactory.createLineChart(
170         "Complexity Line Chart",
171         "Version ID",
172         "Complexity",
173         objDataset,
174         PlotOrientation.VERTICAL,
175         true,
176         true,
177         false
178     );
179     panels[0] = new ChartPanel(objChart);
180     DefaultCategoryDataset objDataset2 = new DefaultCategoryDataset();
181     for(i=0; i<versionlist.size(); i++){
182         double dops = versionlist.get(i).getopsChanges().getMaintenanceActivities();
183         String xaxis = versionlist.get(i).getID();
184         objDataset2.setValue(dops,"Operations", xaxis);
185     }
186     for(i=0; i<versionlist.size(); i++){
187         double dstructs =
versionlist.get(i).getstructsChanges().getMaintenanceActivities();
188         String xaxis = versionlist.get(i).getID();
189         objDataset2.setValue(dstructs,"Structs", xaxis);
190     }
191     JFreeChart objChart2 = ChartFactory.createBarChart(
192         "Maintenance Bar Chart",
193         "Version ID",

```

```

194         "Maintainance",
195         objDataset2,
196         PlotOrientation.VERTICAL,
197         true,
198         true,
199         false
200     );
201     panels[1] = new ChartPanel(objChart2);
202     return panels;
203 }
204
205
206
207 public ChartPanel[] thirdRuleGraphGenerator()
208 {
209     ChartPanel []panels= new ChartPanel[2];
210     DefaultCategoryDataset objDataset = new DefaultCategoryDataset();
211     ArrayList<VersionInfo> versionlist = h.getVersions();
212     int i;
213     for(i=0; i<versionlist.size(); i++){
214         double dops = versionlist.get(i).getopsMetrics().getGrowthRate();
215         String xaxis = versionlist.get(i).getID();
216         objDataset.setValue(dops,"Operations", xaxis);
217     }
218     JFreeChart objChart = ChartFactory.createLineChart(
219         "Growth Rate Line Chart",
220         "Version ID",
221         "Growth Rate",
222         objDataset,
223         PlotOrientation.VERTICAL,
224         true,
225         true,
226         false
227     );
228     panels[0] = new ChartPanel(objChart);
229     DefaultCategoryDataset objDataset2 = new DefaultCategoryDataset();
230     for(i=0; i<versionlist.size(); i++){
231         double dstructs = versionlist.get(i).getstructsMetrics().getGrowthRate();
232         String xaxis = versionlist.get(i).getID();
233         objDataset2.setValue(dstructs,"Structs", xaxis);
234     }
235     JFreeChart objChart2 = ChartFactory.createLineChart(
236         "Growth Rate Bar Chart",
237         "Version ID",
238         "Growth Rate",
239         objDataset2,
240         PlotOrientation.VERTICAL,
241         true,
242         true,
243         false
244     );
245     panels[1] = new ChartPanel(objChart2);
246
247     return panels;
248 }
249
250 public ChartPanel[] fourthRuleGraphGenerator()
251 {
252     ChartPanel []panels= new ChartPanel[2];
253     DefaultCategoryDataset objDataset = new DefaultCategoryDataset();
254     ArrayList<VersionInfo> versionlist = h.getVersions();
255     int i;

```

```

256     for(i=0; i<versionlist.size(); i++){
257         double dops = versionlist.get(i).getopsMetrics().getWorkRate();
258         String xaxis = versionlist.get(i).getID();
259         objDataset.setValue(dops, "Operations", xaxis);
260     }
261     JFreeChart objChart = ChartFactory.createLineChart(
262         "Work Rate Line Chart",
263         "Version ID",
264         "Work Rate",
265         objDataset,
266         PlotOrientation.VERTICAL,
267         true,
268         true,
269         false
270     );
271     panels[0] = new ChartPanel(objChart);
272     DefaultCategoryDataset objDataset2 = new DefaultCategoryDataset();
273     for(i=0; i<versionlist.size(); i++){
274         double dstructs = versionlist.get(i).getstructsMetrics().getWorkRate();
275         String xaxis = versionlist.get(i).getID();
276         objDataset2.setValue(dstructs, "Structs", xaxis);
277     }
278     JFreeChart objChart2 = ChartFactory.createLineChart(
279         "Work Rate Bar Chart",
280         "Version ID",
281         "Work Rate",
282         objDataset2,
283         PlotOrientation.VERTICAL,
284         true,
285         true,
286         false
287     );
288     panels[1] = new ChartPanel(objChart2);
289     return panels;
290 }
291
292 public ChartPanel[] fifthRuleGraphGenerator()
293 {
294     return thirdRuleGraphGenerator();
295 }
296
297 public ChartPanel[] sixthRuleGraphGenerator()
298 {
299
300     ChartPanel []panels= new ChartPanel[2];
301     DefaultCategoryDataset objDataset = new DefaultCategoryDataset();
302     ArrayList<VersionInfo> versionlist = h.getVersions();
303     int i;
304     for(i=0; i<versionlist.size(); i++){
305         double dops = versionlist.get(i).getopsMetrics().getNumber();
306         String xaxis = versionlist.get(i).getID();
307         objDataset.setValue(dops, "Operations", xaxis);
308     }
309     JFreeChart objChart = ChartFactory.createLineChart(
310         "Work Rate Line Chart",
311         "Version ID",
312         "Work Rate",
313         objDataset,
314         PlotOrientation.VERTICAL,
315         true,
316         true,
317         false

```

```

318         );
319         panels[0] = new ChartPanel(objChart);
320         DefaultCategoryDataset objDataset2 = new DefaultCategoryDataset();
321         for(i=0; i<versionlist.size(); i++){
322             double dstructs = versionlist.get(i).getstructsMetrics().getNumber();
323             String xaxis = versionlist.get(i).getID();
324             objDataset2.setValue(dstructs, "Structs", xaxis);
325         }
326         JFreeChart objChart2 = ChartFactory.createLineChart(
327             "Work Rate Bar Chart",
328             "Version ID",
329             "Work Rate",
330             objDataset2,
331             PlotOrientation.VERTICAL,
332             true,
333             true,
334             false
335         );
336         panels[1] = new ChartPanel(objChart2);
337
338         return panels;
339     }
340
341     public ChartPanel[] seventhRuleGraphGenerator()
342     {
343         return null;
344     }
345
346     public ChartPanel[] eighthRuleGraphGenerator()
347     {
348         return null;
349     }
350 }
351

```