

# 知能プログラミング演習II 課題3

グループ 8

29114003 青山周平

2019年11月05日

提出物 rep3

グループ グループ 8

メンバー	学生番号	氏名	貢献度比率
	29114003	青山周平	null
	29114060	後藤拓也	null
	29114116	増田大輝	null
	29114142	湯浅範子	null
	29119016	小中祐希	null

## 1 課題の説明

**必須課題 3-1** セマンティックネットのプログラムを参考に，グループメンバー全員（およびその周辺人物）についてのセマンティックネットを構築せよ．個人レポートには自分のみ（とその周辺）に関するセマンティックネットを示し，グループレポートには全員（とその周辺）に関するセマンティックネットを示せ．

**必須課題 3-2** フレームのプログラムを参考に，自分達の興味分野に関する知識をフレームで表現せよ．その分野の知識を表す上で必須となるスロットが何かを考え，クラスフレームを設計すること．個人レポートには自分が作ったインスタンスフレームのみ（クラスフレームの設計担当者はクラスフレームも）を示し，グループレポートにはクラスフレームおよび全員分のインスタンスフレームを示せ．

**必須課題 3-3** 課題 3-1 または 3-2 で作った知識表現を用いた質問応答システムを作成せよ。なお、ユーザの質問は英語や日本語のような自然言語が望ましいが、難しければ課題 2 で扱ったような変数を含むパターン (クエリー) でも構わない。

**発展課題 3-4** 課題 3-1 または 3-2 で作った知識表現を図として示すためのユーザインターフェース (GUI) を設計し実装せよ。

**発展課題 3-5** 上記 3-3 で作成した質問応答システムを、DBpedia あるいは Wikidata 中の知識を使って質問に答えられるよう、拡張せよ。

## 2 必須課題 3-1

セマンティックネットのプログラムを参考に、グループメンバー全員（およびその周辺人物）についてのセマンティックネットを構築せよ。個人レポートには自分のみ（とその周辺）に関するセマンティックネットを示し、グループレポートには全員（とその周辺）に関するセマンティックネットを示せ。

私（とその周辺）に関するセマンティックネットの構造は、下図の通りである。

図 1: セマンティックネット

## 3 必須課題 3-2

フレームのプログラムを参考に、自分達の興味分野に関する知識をフレームで表現せよ。その分野の知識を表す上で必須となるスロットが何かを考え、クラスフレームを設計すること。個人レポートには自分が作ったインスタンスフレームのみ（クラスフレームの設計担当者はクラスフレームも）を示し、グループレポートにはクラスフレームおよび全員分のインスタンスフレームを示せ。

私が作ったインスタンスフレームは下図のような設計となっている。

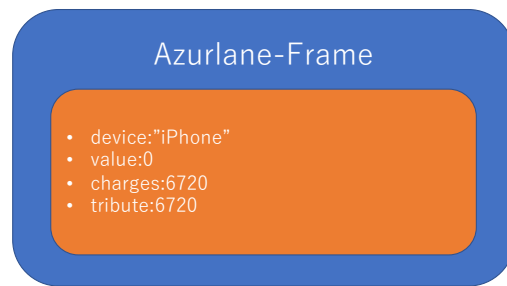


図 2: インスタンスフレーム

## 4 発展課題 3-4

課題 3-1 または 3-2 で作った知識表現を図として示すためのユーザインターフェース (GUI) を設計し実装せよ。

私の担当箇所は，発展課題 3-4 における GUI の Swing を用いた実装である。

### 4.1 手法

セマンティックネットのための GUI を実装するにあたり，以下のような方針を立てた。

1. ノードやリンクのデータを受け取る。
2. ノードを表示するためのパネルを作り，リンクを表示するための描写を設定する。また，パネルをマウスで移動できるようにする。
3. 検索・追加・削除の機能を追加する。

1. に関して，他のプログラムとのデータのやりとりの仲介プログラムを他の班員に作ってもらうことで，作業の分担・効率化を図った。

2. に関して，セマンティックネットのような複雑な図をユーザが自身で最も見やすい表示にするために，マウスによるドラッグでパネルを移動できるような仕様とした。

3. に関して，これらの機能をセマンティックネットとは別のパネルに分けたことで，より構造化された，拡張性・保守性の高い管理しやすいプログラムとなるようにした。

また，セマンティックネットの GUI を元にして，フレームの GUI も作った。

## 4.2 実装

セマンティックネットの GUI に関するプログラム SemNetGUI.java には以下のクラスが含まれる.

- SemNetGUI: メソッド main, クラス MenuPanel を実装した, フレームとメニューバーを実装するためのクラス.
- RelationMap: セマンティックネット全体を管理するパネルを実装するためのクラス.
- NodePanel: 1つのノードに関するパネルを実装するためのクラス.
- LinkPanel: 1つのリンクに関するパネルを実装するためのクラス.

フレームの GUI に関するプログラム FrameGUI.java には以下のクラスが含まれる.

- FrameGUI: メソッド main を実装したフレームを実装するためのクラス.
- FrameMap: 全フレームを管理するパネルを実装するためのクラス.
- FramePanel: 1つのフレームに関するパネルを実装するための抽象クラス.
- ClassFramePanel: クラスフレームのときに用いる FramePanel クラスを継承したクラス.
- InstanceFramePanel: インスタンスフレームのときに用いる FramePanel クラスを継承したクラス.
- LinkPanel: 2つのフレーム間の1つの継承関係に関するパネルを実装するためのクラス.

### 4.2.1 ノードやリンクのデータを受け取る

ノードやリンクの作成や取得には SemanticNet.java を呼び出す必要がある. そこで, その仲介のためのプログラム AccessData.java を他の班員

に作ってもらった。私の担当箇所である SemNetGUI.java からは AccessData.java を介することで SemanticNet.java で必要な処理をより簡単に行えるようにした。

RelationMap クラスのコンストラクタにおける AccessData を用いた処理をソースコード 1 に示す。

ソースコード 1: RelationMap クラスのコンストラクタ

---

```
1      RelationMap(String filename) {
2          sn = new SemanticNet();
3          ad = new AccessData(sn);
4          ad.start(filename); // セマンティックネットの構築
5          ...
6          nodes = new HashMap<>();
7          ArrayList<Node> nodeList = ad.getNodes(); // セマ
              ティックネットからノードの取得
8          ...
9          links = new HashMap<>();
10         ArrayList<Link> linkList = ad.getLinks(); // セマ
              ティックネットからリンクの取得
11         ...
12     }
13 }
```

---

#### 4.2.2 ノードを表示するためのパネルを作り、リンクを表示するための描写を設定する。また、パネルをマウスで移動できるようにする

ノードのパネルの作成には JPanel クラスを継承した NodePanel クラスを実装し、利用した。このパネルはマウスでドラッグできるようにする必要があるため、それを実装するために MouseAdapter クラスの mousePressed メソッドと MouseMotionAdapter クラスの mouseDragged メソッドをオーバーライドすることで、実装した。

また、これらのイベントモデルは全て匿名クラスで実装した。これにより、ソースコードがスッキリし、挙動も分かりやすくすることができた。

NodePanel クラスのコンストラクタにおけるドラック処理をソースコード 2 に示す。

ソースコード 2: NodePanel クラスのコンストラクタ

---

```
1      NodePanel(Node node) {
```

---

```

2      ...
3      addMouseListener(new MouseAdapter() {
4          @Override
5          public void mousePressed(MouseEvent e) {
6              draggedAtX = e.getX();
7              draggedAtY = e.getY();
8          }
9      });
10
11     addMouseMotionListener(new MouseMotionAdapter() {
12         @Override
13         public void mouseDragged(MouseEvent e) {
14             setLocation(e.getX() - draggedAtX +
15                         getLocation().x, e.getY() - draggedAtY
16                         + getLocation().y);
17
18             for (LinkPanel lp : departFromMeLinks) {
19                 lp.update();
20                 lp.repaint();
21             }
22             for (LinkPanel lp : arriveAtMeLinks) {
23                 lp.update();
24                 lp.repaint();
25             }
26         }
27     });
28     ...
29 }

```

---

まず、mousePressed メソッドでは e.getX(), e.getY() メソッドによって、マウスがクリックされた時点での座標が取得される。これを NodePanel クラスのフィールドである int 型変数 draggedAtX, draggedAtY に代入し、保持している。

次に mouseDragged メソッドでは、マウスがクリックされたままドラッグされる度に setLocation メソッドによって自身のインスタンスの座標を更新している。ここで setLocation にて行われている計算式について、x 軸について特筆して述べる。getLocation().x は NodePanel が配置されるコンポーネントにおける座標であるのに対して、e.getX() と draggedAtX は NodePanel 内における座標である。すなわち setLocation メソッドが実行される度に e.getX() の座標は draggedAtX に一致するため、この計算

式で正しく動作することが証明できる。

また、mouseDragged メソッドでは、update メソッドと repaint メソッドの呼び出しが行われている。これにより、ノードに対応するリンクについてもパネルが更新されている。

リンクのパネルについて、ノードのパネルと違い、パネルの移動だけでなく、パネルの拡大縮小・矢印の再描写が求められる。そこで、パネルの座標と大きさの設定を行える setSize メソッド、setSize メソッドを用いて更にラベルの位置も更新する update メソッドを実装した。LinkPanel クラス内のこれらのメソッドをソースコード 3 に示す。

ソースコード 3: setSize メソッド・update メソッド

---

```
1  void setSize() {
2      int lpX = getLeft();
3      int lpY = getTop();
4      int lpWidth = getRight() - lpX;
5      int lpHeight = getBtm() - lpY;
6      setBounds(lpX, lpY, lpWidth, lpHeight);
7  }
8
9  void update() {
10     Rectangle source = tail.getBounds();
11     Rectangle distance = head.getBounds();
12     setShortestDistance(source, distance);
13     setSize();
14
15     int fitX = -(mainPanel.getWidth() / 2);
16     int fitY = -(mainPanel.getHeight() / 2);
17     mainPanel.setLocation((getRight() - getLeft()) /
18         2 + fitX, (getBtm() - getTop()) / 2 + fitY);
19 }
```

---

getLeft, getTop, getRight, getBtm メソッドはパネルの座標を取得するために実装されている。

#### 4.2.3 検索・追加・削除の機能を追加する

内部クラス



#### 4.2.4 セマンティックネットの GUI を元にして、フレームの GUI を作る

FrameGUI.java は、基本的に SemNetGUI.java を元にして作られている。しかし Frame には Node と異なり、クラスフレームやインスタンスフレームといった概念が存在する。これを解決するために、FramePanel クラスを抽象クラスとして、クラスフレームのためのパネルには ClassFramePanel クラス、インスタンスフレームのためのパネルには InstanceFramePanel クラスを実装して利用することで、それぞれの挙動に合わせたパネルを作り分けることができた。

ClassFramePanel クラス、InstanceFramePanel クラスをソースコード 4 に示す。

ソースコード 4: ClassFramePanel クラス・InstanceFramePanel クラス

---

```
1 class ClassFramePanel extends FramePanel {
2     private ArrayList<InstanceFramePanel> list;
3
4     ClassFramePanel(String name, List<String> slotLabels,
5                     List<String> values) {
6         super(name, slotLabels, values);
7         list = new ArrayList<>();
8     }
9
10    void addInstance(InstanceFramePanel ifp) {
11        list.add(ifp);
12    }
13
14 class InstanceFramePanel extends FramePanel {
15     private ClassFramePanel par;
16
17     InstanceFramePanel(String name, ClassFramePanel cfp,
18                       List<String> values) {
19         super(name, new ArrayList(cfp.getSlots().keySet
20                                   ()), values);
21         par = cfp;
22         cfp.addInstance(this);
23     }
24 }
```

---

### 4.3 実行例

UnifyGUI を実行したところ，下図のような画面が得られる．

図 3: 初期状態

#### 4.4 考察

### 5 感想

#### 参考文献

TATSUO IKURA : 『Swing を使ってみよう - Java GUI プログラミング』 <https://www.javadrive.jp/tutorial/> (2019/11/05 アクセス)