Национальный Исследовательский Университет ИТМО
Факультет Программной Инженерии и Компьютерной Техники

Вариант № 86006
Лабораторная работа № 3
по дисциплине
'Программирование'

Выполнил:
Студент группы P3113
***Куперштейн Дмитрий;* : 269359**
Преподаватель:
ПИСЬМАК АЛЕКСЕЙ ЕВГЕНЬЕВИЧ

Санкт-Петербург 2019 г.

Содержание

1 Задание

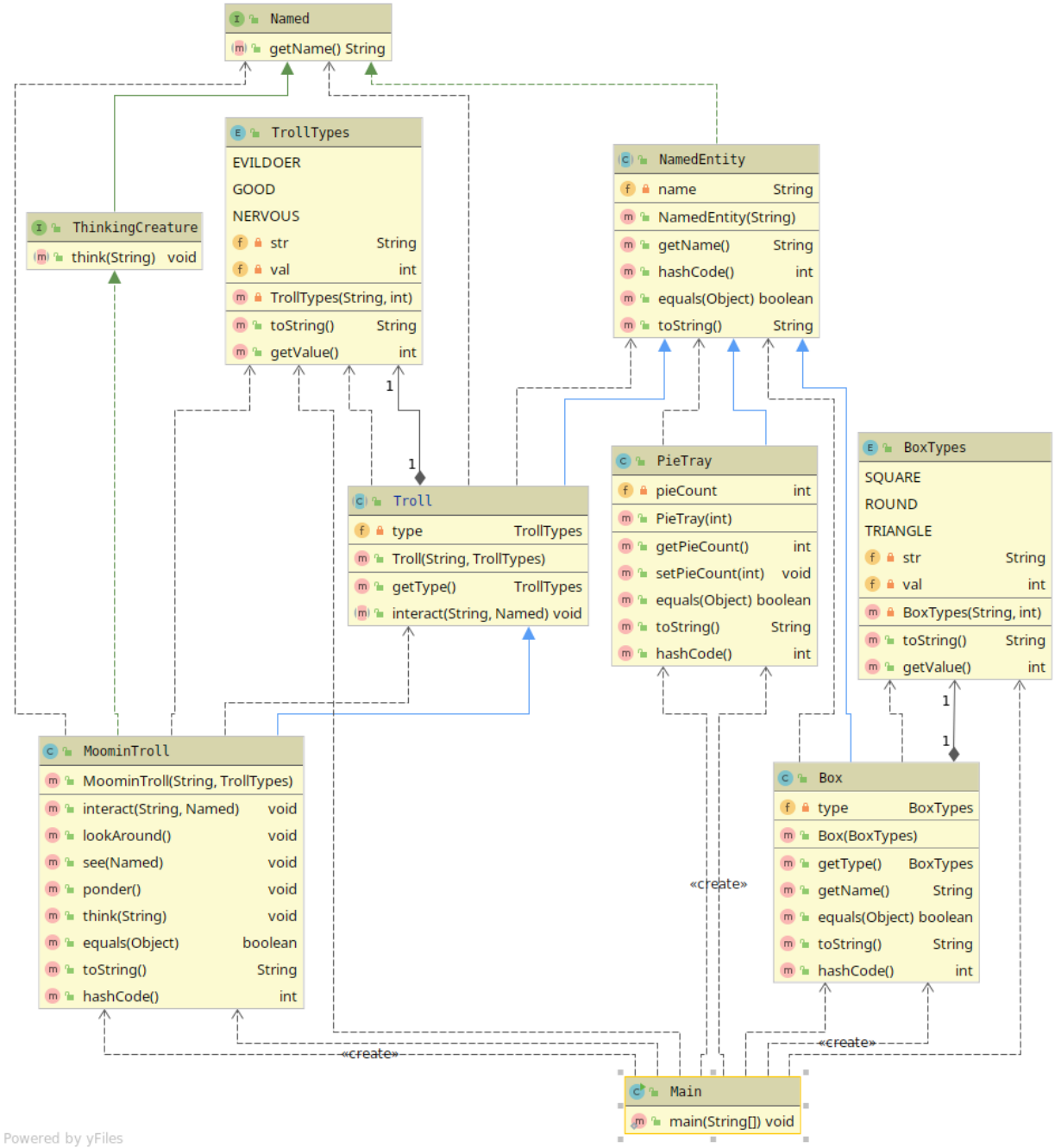
Описание предметной области, по которой должна быть построена объектная модель (для варианта 86006):

> "Страсть как забавно, – подумала она. – Вот уж удивится моя сестра!"Оглядевшись вокруг, она заметила плывшие рядом поднос для пирожков и шкатулку Мумимамы. После недолгого раздумья (хотя на подносе еще оставалось несколько пирожков) она выбрала шкатулку и залезла туда.

Программа должна удовлетворять следующим требованиям:

1. Доработанная модель должна соответствовать принципам SOLID.
2. Программа должна содержать как минимум два интерфейса и один абстрактный класс (номенклатура должна быть согласована с преподавателем).
3. В разработанных классах должны быть переопределены методы `equals()`, `toString()` и `hashCode()`.

2 Диаграмма классов объектной модели



3 Исходный код программы

3.0.1 Main.java

```java
package com.kupp.prog3lab;

public class Main {
  public static void main(String[] args) {
    MoominTroll she = new MoominTroll("She", TrollTypes.GOOD);
    PieTray tray = new PieTray(8);
    Box box = new Box(BoxTypes.ROUND);
    she.think("that's funny");
    she.think("Sister would be surprised");
    she.lookAround();
    she.see(tray);
    she.see(box);
    she.ponder();
    int pieCount = tray.getPieCount();
    if (pieCount > 2) {
      she.interact(String.format("ate %d pies off", pieCount - 2), tray);
      tray.setPieCount(2);
    }
    she.interact("climbed into", box);
  }
}
```

3.0.2 Named.java

```java
package com.kupp.prog3lab;

public interface Named {
  String getName();
}
```

3.0.3 NamedEntity.java

```java
package com.kupp.prog3lab;

public abstract class NamedEntity implements Named {
  private String name;

  public NamedEntity(String name) {
    this.name = name;
  }

  public String getName() {
    return this.name;
  }

  @Override
  public int hashCode() {
    return getName().hashCode();
  }

  @Override
  public boolean equals(Object o) {
```

```
    if (!(o instanceof NamedEntity)) {
      return false;
    }
    NamedEntity other = (NamedEntity) o;
    return other.getName().equals(this.getName());
  }

  @Override
  public String toString() {
    return String.format("NamedEntity with name \"%s\"", this.getName());
  }
}
```

3.0.4 ThinkingCreature.java

```
package com.kupp.prog3lab;

public interface ThinkingCreature extends Named {
  void think(String about);
}
```

3.0.5 Troll.java

```
package com.kupp.prog3lab;

public abstract class Troll extends NamedEntity {
  private TrollTypes type;

  public Troll(String name, TrollTypes type) {
    super(name);
    this.type = type;
  }

  public TrollTypes getType() {
    return type;
  }

  public abstract void interact(String action, Named named);
}
```

3.0.6 MoominTroll.java

```
package com.kupp.prog3lab;

public class MoominTroll extends Troll implements ThinkingCreature {
  public MoominTroll(String name, TrollTypes type) {
    super(name, type);
  }

  public void interact(String action, Named named) {
    System.out.printf("%s %s %s.\n", this.getName(), action, named.getName());
  }

  public void lookAround() {
    System.out.printf("%s look around.\n", this.getName());
  }
```

```
15
16    public void see(Named named) {
17      System.out.printf("%s see %s.\n", this.getName(), named.getName());
18    }
19
20    public void ponder() {
21      System.out.printf("%s is thinking about it.\n", this.getName());
22    }
23
24    public void think(String about) {
25      System.out.printf("%s think that %s.\n", this.getName(), about);
26    }
27
28    @Override
29    public boolean equals(Object o) {
30      if (!(o instanceof MoominTroll)) {
31        return false;
32      }
33      MoominTroll other = (MoominTroll) o;
34      return other.getName().equals(this.getName())
35              && other.getType().equals(this.getType());
36    }
37
38    @Override
39    public String toString() {
40      return String.format("MoominTroll with name \"%s\"" +
41              " & \"%s\" type", this.getName(), this.getType().toString());
42    }
43
44    @Override
45    public int hashCode() {
46      return super.hashCode() + this.getType().getValue();
47    }
48  }
```

3.0.7 PieTray.java

```
1   package com.kupp.prog3lab;
2
3   public class PieTray extends NamedEntity {
4     private int pieCount;
5     public PieTray(int pieCount) {
6       super("pie tray");
7       this.pieCount = pieCount;
8     }
9
10    public int getPieCount() {
11      return pieCount;
12    }
13
14    public void setPieCount(int pieCount) {
15      if (pieCount >= 0) {
16        this.pieCount = pieCount;
17      }
18    }
19
20    @Override
21    public boolean equals(Object o) {
22      if (!(o instanceof PieTray)) {
```

```
23        return false;
24      }
25      PieTray other = (PieTray)o;
26      return other.getPieCount() == this.getPieCount();
27    }
28
29    @Override
30    public String toString() {
31      return String.format("PieTray with %d pies", this.getPieCount());
32    }
33
34    @Override
35    public int hashCode() {
36      return this.getPieCount();
37    }
38  }
```

3.0.8 Box.java

```
1   package com.kupp.prog3lab;
2
3   public class Box extends NamedEntity {
4     private BoxTypes type;
5
6     public Box(BoxTypes type) {
7       super("box");
8       this.type = type;
9     }
10
11    public BoxTypes getType() {
12      return type;
13    }
14
15    @Override
16    public String getName() {
17      return String.format("%s %s", type.toString(), super.getName());
18    }
19
20    @Override
21    public boolean equals(Object o) {
22      if (!(o instanceof Box)) {
23        return false;
24      }
25      Box other = (Box) o;
26      return other.getName().equals(this.getName());
27    }
28
29    @Override
30    public String toString() {
31      return this.getName();
32    }
33
34    @Override
35    public int hashCode() {
36      return this.getName().hashCode();
37    }
38  }
```

3.0.9 TrollTypes.java

```java
package com.kupp.prog3lab;

public enum TrollTypes {
  EVILDOER("evildoer", 0),
  GOOD("good", 1),
  NERVOUS("nervous", 2);

  private String str;
  private int val;
  TrollTypes(String str, int val) {
    this.str = str;
    this.val = val;
  }

  @Override
  public String toString() {
    return this.str;
  }

  public int getValue() {
    return this.val;
  }
}
```

3.0.10 BoxTypes.java

```java
package com.kupp.prog3lab;

public enum BoxTypes {
  SQUARE("square", 0),
  ROUND("round", 1),
  TRIANGLE("triangle", 2);

  private String str;
  private int val;

  BoxTypes(String str, int val) {
    this.str = str;
    this.val = val;
  }

  @Override
  public String toString() {
    return this.str;
  }

  public int getValue() {
    return this.val;
  }
}
```

4 Результат работы программы

```
She think that that's funny.
She think that Sister would be surprised.
She look around.
She see pie tray.
She see round box.
She is thinking about it.
She ate 6 pies off pie tray.
She climbed into round box.
```

5 Вывод

В ходе этой лабораторной работы я поломал голову над реализацией абстрактной объектной модели по тексту, это оказалось непростой задачей для меня, так как пришлось достраивать модель текста для соответствия работы требованиям. Также я научился вписывать в enum в Java дополнительные методы.