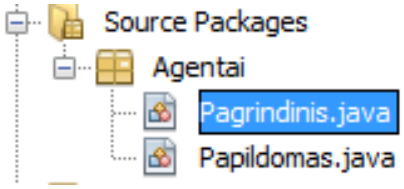
Trečias laboratorinis darbas Nr.3

Darbo tikslas:

- Sukurti pagrindinį agentą, kuris galėtų sukurti naujus konteinerius ir juose veikiančius agentus.
- Agentai tarpusavyje bendrauja ACL **(FIPA Propose Interaction Protokolu)**.

1. Atsidarykite antrą laboratorinį darbą, kuriame susikurkite Pagrindinį ir papildomą agentą.



2. Papildykite Pagrindinį agentą

2.1. Reiks įtraukti papildomus metodus:

- ContainerController Sukurti konteinerį (String Hostas, String Portas, String Vardas)

– šis metodas grąžina suformuotą konteinerio valdiklį (ContainerController). Metodo parametrai **host, port** ir **name** yra reikalingi konteinerio valdiklio (ContainerController) sukūrimui.

- SukurtiAgenta konteineryje (String Pav, ContainerController Konteineris).

Naudojamų funkcijų paaiškinimas:

- `Profile.setParameter(parameter, value)` – konteinerio parametro “parameter” nustatymas į “value” reikšmę;
- `Runtime.createAgentContainer(Profile)` – konteinerio su “Profile” parametrais sukūrimas;
- `SukurtiAgenta(String name, ContainerController container)` – metodas sukuria agentą vardu “name” konteineryje “container”.

2.2. Norint sukurti agentus naujame konteineryje pirmausiai reikia sukurti naują konteinerį, po to jame sukurti norimus agentus. Pirma sukuriam konteinerio vardą (name = “Klase”):

```
String Vardas;
Vardas = "Klase";
```

2.3. Sukuriam konteinerį **cc**, jam perduodant konteinerio vardą, bei kitus reikalingus parametrus. **Host** ir **Port** kintamieji gaunami nuskaitant pagrindinio agento, esančio pagrindiniame konteineryje (main-container), kintamuosius MAIN_PORT, MAIN_HOST, kadangi kuriami konteineriai turi būti susieti su pagrindiniu konteineriu (main-container).

```
ContainerController cc = SukutriKonteineri
   (myAgent.getProperty(Profile.MAIN_HOST, null),
    myAgent.getProperty(Profile.MAIN_HOST, null), Vardas);
SukurtiAgentaKonteineryje("Papildomas",cc);
```

2.4. Papildykite programos kodą **protected void setup()** dalyje**:**

```
private ContainerController SukutriKonteineri(String Hostas,String Portas, String Vardas)
{

    Runtime rt = Runtime.instance();
    Profile p = new ProfileImpl();
    p.setParameter(Profile.MAIN_HOST, Hostas);
    p.setParameter(Profile.MAIN_PORT, Portas);
    p.setParameter(Profile.CONTAINER_NAME, Vardas);
    ContainerController cc = rt.createAgentContainer(p);
    return cc;
}
```

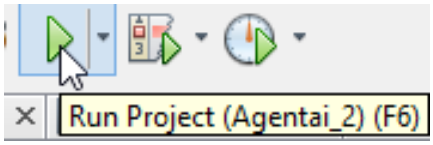
2.5. Papildykite programos kodą **protected void setup()** dalyje**:**

```
private void SukurtiAgentaKonteineryje(String Pav,ContainerController Konteineris)
{
    String Pavadinimas = Pav;
    try
    {
       AC = Konteineris.createNewAgent( Pavadinimas, pack +"." + Pavadinimas ,null );
       AC.start();
       //Programos kodas
    }
    catch(Exception any)
    {
        any.printStackTrace();
    }
}
```

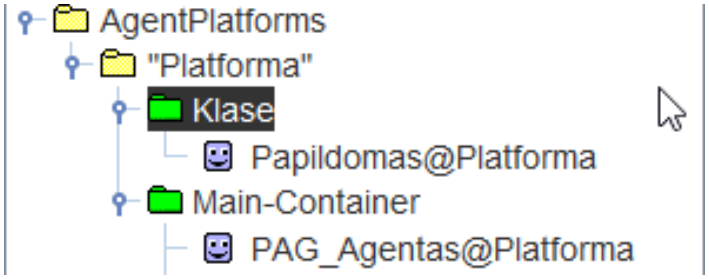
2.6. Papildykite programos kodą **nurodytoje dalyje** taip, kad **atspausdintų“**

- Pagrindinio agento vardą;
- Pagrindinio agento sukurto agento vardą;
- Konteinerio pavadinimą.

2.7. Paleiskite programą

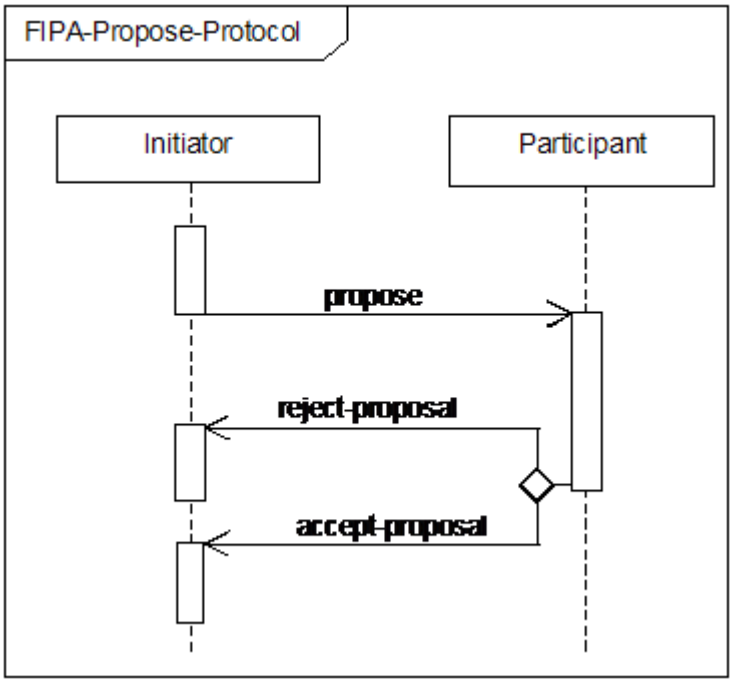


2.8. Iš programos lango matome, kad papildomas agentas sukurtas naujame konteineryje **Klase**



3. Agentų bendravimas ACL (FIPA Propose Interaction Protokolu)

3.1. **FIPA Propose Interaction protokolas**



Daugiau informacijos apie FIPA: http://www.fipa.org/specs/fipa00036/SC00036H.html

Apir propose, reject-proposal ir accept-proposal:

http://www.fipa.org/specs/fipa00037/SC00037J.html

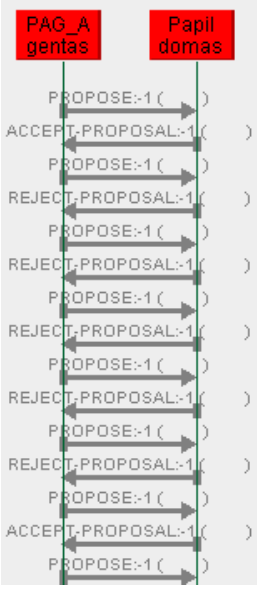
3.2. FIPA Papildykite **pagrindinio** agento programos kodą:

```
addBehaviour(new TickerBehaviour(this, 2000)
{
    protected void onTick()
    {
      ACLMessage cfp;
      cfp = new ACLMessage(ACLMessage.PROPOSE);
      cfp.addReceiver(new AID("Papildomas", AID.ISLOCALNAME));
      cfp.setContent(Argumentas);
      myAgent.send(cfp);
    }
});
```

3.3. Savarankiškai papildykite **pagrindinio** agento programos kodą taip, kad a*rgumentas* būtų skaitomas iš argumentų eilutės.

3.4. Papildykite **Papildomas** agento kodą taip, kad agentas priimtų arba atmestų pasiūlymą su 50% tikimybe.

3.5. Agentų bendravimas pavyzdinis langas pateiktas žemiau:



Punktų realizacijos

2.6 Punkto realizacija:

```
private void SukurtiAgentaKonteineryje(String Pav,ContainerController Konteineris)
{
    String Pavadinimas = Pav;
    try
    {
       AC = Konteineris.createNewAgent( Pavadinimas, pack +"." + Pavadinimas ,null );
       AC.start();
       System.out.println("----------------------------------------");
       System.out.println("Pagrindinis agentas : " + getLocalName());
       System.out.println("Pagrindinio agento sukurtas agentas : " + Pavadinimas);
       System.out.println("Konteineris : " + Konteineris.getContainerName());
       System.out.println("----------------------------------------");

    }
    catch(Exception any)
    {
        any.printStackTrace();
    }
}
```

3.3 Punkto realizacija:

```
Object[] args = getArguments();
Argumentas = (String) args[0];
```

3.4 Punkto realizacija:

```
if (vardas.equals("PAG_Agentas"))
    {
    turinys = zinute.getContent();
    ACLMessage atsakymas;
    int randomNum = 0 + (int)(Math.random() * 2);
    String ats = Integer.toString(randomNum);
    System.out.println(ats);
     switch (randomNum)
     {
        case 0:
    atsakymas = new ACLMessage(ACLMessage.REJECT_PROPOSAL);
    atsakymas.addReceiver(new AID(vardas, AID.ISLOCALNAME));

    System.out.println("Papildomas agentas atmeta pasiulima");
    System.out.println(ats);
    atsakymas.setContent("Papildomas agentas atmeta pasiulima");
    send(atsakymas);
       break;

        case 1:
    atsakymas = new ACLMessage(ACLMessage.ACCEPT_PROPOSAL);
    atsakymas.addReceiver(new AID(vardas, AID.ISLOCALNAME));

     System.out.println("Papildomas agentas priima pasiulima");
    System.out.println(ats);
    atsakymas.setContent("Papildomas agentas priima pasiulima");
    send(atsakymas);
        break;
     }
```