Szoftvertechnológia

Házi feladat

Tóth Péter

NEPTUN

BME IIT, 2016

# Feladatleírás

[A feladat max. 1000 karakteres részletezettségű magyar nyelvű leírása.]

# Követelmények

### Funkcionális követelmények

[Az alábbi táblázat kitöltésével készítendő.]

|  |  |  |  |
| --- | --- | --- | --- |
| **Azonosító** | **Leírás** | **Use-case** | **Komment** |
|  |  |  |  |

### Egyéb követelmények

[Nem funkcionális követelmények, korlátozások, a táblázat kitöltésével]

|  |  |  |
| --- | --- | --- |
| **Azonosító** | **Leírás** | **Komment** |
|  |  |  |

# Use-case leírás

## Use-case leírások

[Minden use-case-hez külön. 3-5 use-case. Legyen 1 use-case, amit legalább egy másik use-case include-ol!]

|  |  |
| --- | --- |
| **Use-case neve** |  |
| **Rövid leírás** |  |
| **Aktorok** |  |
| **Forgatókönyv** |  |

|  |  |
| --- | --- |
| **Use-case neve** |  |
| **Rövid leírás** |  |
| **Aktorok** |  |
| **Forgatókönyv** |  |

|  |  |
| --- | --- |
| **Use-case neve** |  |
| **Rövid leírás** |  |
| **Aktorok** |  |
| **Forgatókönyv** |  |

|  |  |
| --- | --- |
| **Use-case neve** |  |
| **Rövid leírás** |  |
| **Aktorok** |  |
| **Forgatókönyv** |  |

|  |  |
| --- | --- |
| **Use-case neve** |  |
| **Rövid leírás** |  |
| **Aktorok** |  |
| **Forgatókönyv** |  |

|  |  |
| --- | --- |
| **Use-case neve** |  |
| **Rövid leírás** |  |
| **Aktorok** |  |
| **Forgatókönyv** |  |

## Use-case diagram

# Statikus leírás

[Minden, a feladatban szereplő osztály (interfész) rövid, egy bekezdés hosszú ismertetése. Az osztályok száma 10 és 20 között legyen.]

interface **HasInput**:

Bármilyen osztály, ami képes Csomagot fogadni, az innen örökölt **insert()** metódussal tudja ezt megvalósítani. A csatlakoztatás folyamatát kezdeményezheti **connectToOutput()**

interface **HasOutput**:

Tárol egy referenciát a rákapcsolt HasInput-ról, ennek az insert() függvényét hívja, amikor a kimenetén van egy csomag.

abstract class **TransportObject** implements **HasInput**, **HasOutput**:

Általános csomagátviteli feladatokat valósít meg, definiálja a csatlakoztatáshoz, a doboz továbbadásához szükséges függvényeket, alapértelmezetten a **Ground**-ot használva. A legtöbb esetben ebből érdemes származtatni csomagátvitellel foglalkozó objektumokat..

class RGBColor:

meghatároz egy színt a 4 tárolt byte-ból RGBA szerint

class Event:

egy eseményt határoz meg, tárol egy EvType és egy EvData mezőt

enum EvType:

meghatározza az Event típusát, azaz, hogy a data referenciáját milyen típussá kell castolni, hogy hozzáférjünk

abstract class EventData:

egy esemény részletei, különböző események adatstruktúrái ebből származnak

interface Observable: Event-eket képes dobni Observer pattern szerint a fireEvent(Event) függvénnyel

interface Observer: képes figyelni Observable osztályokat a listenTo(Observable) függvénnyel

ha számára érdekes Event keletkezik, akkor az onEvent(Event) függvényben lesz képes észlelni ezt

class Timer implements Observable:

időzítőt valósít meg, ha valami változtatja az állapotát az idő múlásával, akkor ennek az Event-jeire kell feliratkoznia, hogy megtehesse

## Osztályok leírása

### Osztály1

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy mondat arról, hogy a metódus mit csinál.]

* **int foo(Osztály3 o1, Osztály4 o2)**: metódus leírása
* **int bar(Osztály5 o1)**: metódus leírása

### Osztály2

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy mondat arról, hogy a metódus mit csinál.]

* **int foo(Osztály3 o1, Osztály4 o2)**: metódus leírása
* **int bar(Osztály5 o1)**: metódus leírása

## Osztálydiagram

# Dinamika leírása

[Inicializálásra, use-case-ekre. Konzisztens kell legyen az előző ponttal. Minden metódus, ami ott szerepel, fel kell tűnjön valamelyik szekvenciában. Minden metódusnak, ami szekvenciában szerepel, szerepelnie kell a valamelyik osztálydiagramon.]

## [Use-case neve1] szekvenciája

## [Use-case neve2] szekvenciája

# Állapotmodell

[Csak azokhoz az osztályokhoz, ahol van értelme. Egyetlen állapotból álló state-chartok ne szerepeljenek. Legalább egy osztályhoz tartozzon state-chart.]

## [Osztály1] állapotgépe

# Tevékenységnapló

[A referencia a leírásban szereplő tevékenységgel érintett alpontot jelöli.]

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Leírás** | **Referencia** |
| 2015.11.13. 13:00 | 2 óra | Class Diagram tervezése (félkész)  WhiteStarUML-ről megállapítás, hogy egy hulladék |  |
| 2015.11.21. 00:00 | 1 óra | ClassDiagram kiegészítés, javítgatás |  |
| 2015.11.21. 16:00 | 3 óra | ClassDiagram  befejezés, ellenőrzés, rendezés |  |
| 2015.11.22. 1:10 |  | Class leírások elkészítése |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Egyéb

A felhasznált modellező eszköz: *WhiteStarUML-lel kezdve, utána StarUML*

Egyéb felhasznált eszköz és a használat célja: *XXXXXX*

A feladat megoldására fordított teljes munkaidő órában: *XX*