# Analízis modell kidolgozása

## Objektum katalógus

[Minden, a feladatban szereplő objektum rövid, egy-két bekezdés hosszú ismertetése. Meg kell jelenjen minden objektumhoz, hogy mi a felelőssége. Informális leírás, ezért nem szabad még foglalkozni az örökléssel, az interfészekkel, az absztrakt osztályokkal, a segédosztályokkal.]

### Action

### Álmos

### Breakable

### Control

### Csoki

### Entry

### Exit

### Field

### Fotel

### Ijedős

### Játék

### Orángután

### Panda

### Player

### Szekrény

### Ugrálós

### Unit

[Felelősség informális leírása]

[Az objektumkatalógus alapján kiindulva kell megalkotni az objektumorientált analízis modellt. A 3.2, 3.3, 3.4, 3.5 alfejezetek ugyanannak a modellnek a különböző nézetei, ezért egyidőben, egymással összefüggésben készülnek. Megtörténik az objektumkatalógusban tárgyalt objektumok felelősségének formalizálása osztályokká, attribútumokká, metódusokká . Csak publikus metódusok szerepelhetnek. Megjelennek az interfészek, az öröklés, az absztrakt osztályok. Segédosztályokra még nincs szükség.]

## Statikus struktúra diagramok

[Az objektum katalógusban szereplő objektumokat megvalósító osztályok kapcsolatait és publikus metódusait bemutató osztálydiagram(ok). Tipikus hibalehetőségek: csillag-topológia, szigetek.]

## Osztályok leírása

[A diagramon szereplő osztályok bemutatása. Az osztályok ABC sorrendben kövessék egymást. Interfészek esetén az Interfészek, Attribútumok pontok kimaradnak.]

### Action

#### Felelősség

Az interface a tárgyak körében játszódó visitor pattenhez kapcsolódó függvényeket foglalja össze.

#### Ősosztályok

Interface

#### Metódusok

* **void action(cs:csoki)**: csokiautomatától érkező hívást kezel
* **void action(j:jatek)**: játékgéptől érkező hívást kezel
* **void action(f**:**fotel):** foteltől érkező hívást kezel

### Álmos

#### Felelősség

Az osztály az álmos pandák viselkedéséért felelős

#### Ősosztályok

Object 🡪 Unit 🡪 Panda

#### Interfészek

Az (közvetve) megvalósítja az Action interfészt

#### Attribútumok

**-**

#### Metódusok

* **void action(Fotel f):** Ha az álmos panda fotel közelébe lép, beleül a fotelbe
* a fentieken kívül getter és setter funkciók
* a többi metódusát a pandától örökli

### Breakable

#### Felelősség

Az osztály a törékeny csempék működését valósítja meg

#### Ősosztályok

Object 🡪 Field

#### Interfészek

Nem valósít meg interface-t

#### Attribútumok

* **int lifeTime:** A csempe élettertamát jelöli, azaz hogy az adott csempére hányszor lehet még rálépni úgy, hogy ne essünk le.

#### Metódusok

* **void stepped(Orángután o):** Azt az setet kezeli, mikor a csempére *o* orángután rálépett
* **void stepped(Panda p):** Azt az setet kezeli, mikor a csempére *p* panda rálépett
* a fentieken kívül getter és setter funkciók
* a többi metódusát a Field osztálytól örökli

### Control

#### Felelősség

Az osztály felel a karakterek mozagtásáért, a játék inicializálásáért, inditásáért, lezárásáért. Hídként szolgál a játék és a játékosok között.

#### Ősosztályok

Object 🡪 Control

#### Interfészek

Az oszály nem valósít meg interfészt.

#### Attribútumok

* **-pandas: Panda**: a játékban szereplő Panda objektumok listája
* **-orans: Orangutan**: a játékban szereplő orángutánok listája
* **-player: Player**: a játékosok listája
* **-entryField: Field:** a bejárat referenciája
* **-map: Field:** a csempék listája

#### Metódusok

* **void move(Unit u, Field f)**: az karaktert a megadott mezőre lépteti
* **viod init()**: inicializálja a játékot
* **void start()**: a játék indítása
* **void end()**: a játék vége
* a fentieken kívül getter és setter funkciók

### Csoki

#### Felelősség

Az osztály leírja annak a csempének a működését, amelyen csokiautomata áll.

#### Ősosztályok

Object 🡪 Field

#### Interfészek

Az oszály nem valósít meg interfészt.

#### Attribútumok

#### Metódusok

* **void trigger():** A csokiautomata sípolásáért felelős függvény
* a többi metódusát a Field osztálytól örökli
* a fentieken kívül getter és setter funkciók

### Orangutan

#### Felelősség

Az orángután karakterek mozgása és rá vonatkozó megkötések betartása.

#### Ősosztályok

Object 🡪 Unit

#### Interfészek

Action

#### Attribútumok

* **Hand1: Unit:** A karakter egyik kezében tartott Unitot tárolja.
* **Hand2: Unit:** A karakter másik kezében tartott Unitot tárolja.

A **Hand2** megegyezés alapján kötelezően Pandát tart, a **Hand1** kötelezően null.

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

* **void move(f:field):** mozgatja az orángutánt az f-fieldre.

**Ezen kívül örökölt metódusai, valamint get-set metódusai léteznek.**

### Panda

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

Az panda karakterek mozgása és rá vonatkozó megkötések betartása

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

Object 🡪 Unit

#### Interfészek

[Mely interfészeket valósítja meg.]

Action

#### Attribútumok

[Milyen attribútumai vannak]

**Csak örökölt attribútumai vannak.**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

* **void move(f:field):** mozgatja az pandát az f-fieldre.
* **void grab(u:unit):** a kapott unit-ot, beállítja a Hand1-re ha az null, ha nem akkor a másik kezébe(Hand2). Amennyiben mindkét keze foglalt (nem null) akkor a kérést továbbítja a következő panda felé.

A **Hand2** megegyezés alapján kötelezően Pandát tart, a **Hand1** kötelezően null.

### Player

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Szekrény

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Ugrálós

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Unit

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Panda

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Panda

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Panda

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Panda

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Panda

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**
* **attribútum2**

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

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

### Unit

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés.]

Az osztály a pályán létező és mozgó/mozgatható karakterek egységbe zárásáért felel.

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

Object

#### Interfészek

[Mely interfészeket valósítja meg.]

Action

#### Attribútumok

[Milyen attribútumai vannak]

* **Hand1: Unit:** A karakter egyik kezében tartott Unitot tárolja.
* **Hand2: Unit:** A karakter másik kezében tartott Unitot tárolja.

#### Metódusok

[Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál. Hiba, ha egy osztálynak nincs egyetlen metódusa sem.]

* **void move(f:Field):** Egy megadott mezőre mozgatja a karaktert, tényleges mozgatás nem itt történik.
* **void grab(u:Unit)**: a karakter megfog egy másik karaktert.
* **void exit():** akkor történik, ha egy karakter a kilépési mezőre lép.
* **void die():** a karakter destruktora.
* **release(u:unit):** elengedi a megfelelő kezéből a karaktert
* a fentieken kívül getter és setter funkciók

## Szekvencia diagramok

[Inicializálásra, use-case-ekre, belső működésre. Konzisztens kell legyen az előző alfejezettel. Minden metódus, ami ott szerepel, fel kell tűnjön valamelyik szekvenciában. Minden metódusnak, ami szekvenciában szerepel, szereplnie kell a valamelyik osztálydiagramon. Fontos, hogy az aktor use-case-eitől bármelyik diagramhoz el lehessen jutni a metódushívások követésével.]

## State-chartok

[Csak azokhoz az osztályokhoz, ahol van értelme. Egyetlen állapotból álló state-chartok ne szerepeljenek. A játék működését bemutató state-chart-ot készíteni tilos.]

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2010.03.21. 18:00 | 2,5 óra | Horváth  Németh  Tóth  Oláh | Értekezlet.  Döntés: Horváth elkészíti az osztálydiagramot, Oláh a use-case leírásokat. |
| 2010.03.23. 23:00 | 5 óra | Németh | Tevékenység: Németh implementálja a tesztelő programokat. |
| … | … | … | … |