# Grafikus felület specifikációja

## A grafikus interfész

[A menürendszer, a kezelői felület grafikus képe. A grafikus felület megjelenését, a használt ikonokat, stb screenshot-szerű képekkel kell bemutatni. Az építészetben ez a homlokzati terv.]

## A grafikus rendszer architektúrája

[A felület működésének elve, a grafikus rendszer architektúrája (struktúra diagramok). A struktúra diagramokon a prototípus azon és csak azon osztályainak is szerepelnie kell, amelyekhez a grafikus felületet létrehozó osztályok kapcsolódnak.]

### A felület működési elve

[Le kell írni, hogy a grafikai megjelenésért felelős osztályok, objektumok hogyan kapcsolódnak a meglevő rendszerhez, a megjelenítés során mi volt az alapelv. Törekedni kell az MVC megvalósításra. Alapelvek lehetnek: **push** alapú: a modell értesíti a felületet, hogy változott; **pull** alapú: a felület kérdezi le a modellt, hogy változott-e; **kevert**: a kettő kombinációja.]

A grafikus osztályok példányait factory-k hozzák létre a modellbeli példányok létrehozása során. Minden megjelenítendő modellbeli osztályhoz tartozik egy factory, aki implementálja az adott osztályhoz tartozó abstract factory-t. A modellbeli példányok létrejövésekor a konkrét factory implementáció fogja létrehozni a grafikus felületet reprezentáló osztályok példányait. Mivel mi (Java) Swingben dolgozunk minden megjeleníthető objektumhoz tartozni fog egy SwingObjectFactory és egy SwingObject.

Miután létrejött a view-beli példány is, a modellbeli és view-beli példányokat felvesszük egy map-ba, amit a ViewRepository osztály tartalmaz. A map minden megjeleníthető osztály modell és view párjait tartalmazza. A grafikus felület frissítése a ViewRepository segítségével fog megvalósulni, push alapon.

Amikor valamelyik modellbeli objektum olyan akciót végez, ami miatt a grafikus felületet frissíteni kéne, akkor megívja a ViewRepository-n az updateObject-et és átadja saját magát paraméterként. A ViewRepository ezután kikeresi a mapból a grafikus párját és meghívja rajta az ő update függvényét, amivel frissül a felület.

Összefésült Zsomboréval ez lesz majd a végleges:

A modell elemei nem tudnak semmit a view elemeiről, de a view osztályai képesek olvasni a hozzájuk tartozó modell osztály view-nak átadott interfészét. Minden view osztály implementálja az updateable interfészt, aminek van egy update tagfüggvénye. Ez újrarajzolja az adott view osztályt, a hozzácsatolt modell osztály adatai alapján.

Minden megjelenítendő modellbeli osztályhoz tartozik egy factory, aki implementálja az adott osztályhoz tartozó abstract factory-t. A modellbeli példányok létrejövésekor a konkrét factory implementáció fogja létrehozni a grafikus felületet reprezentáló osztályok példányait. Mivel mi (Java) Swingben dolgozunk minden megjeleníthető objektumhoz tartozni fog egy SwingObject és egy SwingObjectFactory.

A modell és a view kapcsolatát a „ViewRepository” osztály tartja fenn. Ez az osztály tartalmaz magában egy statikus Hash táblát, amelynek a kulcsa egy objektum és az értéke a hozzá tartozó view osztály. A modell összekötése a nézettel a „ViewRepository” bind metódusával történik, ami automatikusan lefut, amint a view példánya létrejön.

Amikor egy view osztály frissítését szeretnénk kérni, akkor azt a modell osztály kezdeményezi. Ezt a „ViewRepository” updateObject tagfüggvényével teheti meg, ahol saját magát adja át, mint argumentum és utána a „ViewRepositry” meghívja a hozzá tartozó view update metódusát.

### A felület osztály-struktúrája

[Osztálydiagram. Minden új osztály, és azon régiek, akik az újakhoz közvetlenül kapcsolódnak.]

## A grafikus objektumok felsorolása

[Az új osztályok felsorolása. Az régi osztályok közül azoknak a felsorolása, ahol változás volt. Ezek esetén csak a változásokat kell leírni.]

### Osztály1

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés. Ha szükséges, akkor state-chart is.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**: attribútum jellemzése: mire való, láthatósága (UML jelöléssel), típusa
* **attribútum2**: attribútum jellemzése: mire való, láthatósága (UML jelöléssel), típusa

#### Metódusok

[Milyen publikus, protected és privát metódusokkal rendelkezik. Metódusonként precíz leírás, ha szükséges, activity diagram is a metódusban megvalósítandó algoritmusról.]

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

### Osztály2

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés. Ha szükséges, akkor state-chart is.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**: attribútum jellemzése: mire való, láthatósága (UML jelöléssel), típusa
* **attribútum2**: attribútum jellemzése: mire való, láthatósága (UML jelöléssel), típusa

#### Metódusok

[Milyen publikus, protected és privát metódusokkal rendelkezik. Metódusonként precíz leírás, ha szükséges, activity diagram is a metódusban megvalósítandó algoritmusról.]

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

## Kapcsolat az alkalmazói rendszerrel

[Szekvencia-diagramokon ábrázolni kell a grafikus rendszer működését. Konzisztens kell legyen az előző alfejezetekkel. Minden metódus, ami ott szerepel, fel kell tűnjön valamelyik szekvenciában. Minden metódusnak, ami szekvenciában szerepel, szerepelnie kell a valamelyik osztálydiagramon.]

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2025.05.01 11:00 | 1 óra 30 perc | Guzmics | Tevékenység:  Felhasználói felület frissítésének kigondolása |
| 2025.05.03 12:00 | 30 perc | Guzmics | Tevékenység:  A felület működésének elvének kezdetleges leírása |