

SmartPaint

Okos rajzprogram - Szoftverarchitektúrák házi feladat

Készítették:

Hegedűs Tamás László

Dusza Andrea

# Specifikáció

## Feladatkiírás

A feladat célja egy olyan rajzprogram létrehozása, mely pixel halmazok formájában adott foltok szerkesztésére, valamint ezeket manipuláló algoritmusok futtatására alkalmas. A rajzprogramban ezeket a foltokat lehet átrajzolni (pixelek törlése vagy hozzáadása), valamint ezeken a foltokon lehet egyszerű műveleteket lefuttatni (például kettő összeolvasztása, nem teljesen körbevett pixelek törlése). (A cél nem bonyolult képmorfológiai műveletek megvalósítása, hanem az ezek befogadására képes, kényelmesen bővíthető (transzformáció plug-in-eket támogató) architektúra kialakítása.)

A program alapvető funkciói:

- Kép betöltése

- Egy "kép -> ponthalmazok halmaza" művelet lefuttatása (a foltok megkeresése)

- Egy "folthalmaz -> folthalmaz" művelet paraméterezése és lefuttatása

- Folthalmaz elmentése, visszatöltése

- Foltok kijelölése

- Foltokból pixelek törlése (radír) vagy hozzáadása (ecset)

Preferált platform: Qt C++ vagy WPF.

## Részletes leírás

A rajzprogram .png, .jpg, és .bmp formátumú nyers képek, illetve az általunk definiált .spt kiterjesztésű SmartPaint projektfájlok (folthalmaz-fájlok) kezelésére lesz képes. Az alkalmazás indításkor azonnal azt kéri, hogy hozzunk létre egy új projektet, vagy a fájlrendszerből töltsünk be egy meglévőt. Ezután a projekten az alábbi műveleteket végezhetjük:

Meglévő foltokon különböző transzformációk végzése

Foltok importálása a projektbe egy másik, már létező projekt fájlból

Nyers kép betöltése által új foltok létrehozása

A projekt foltjainak exportálása .png formátumú képfájlba

Az alkalmazás alapvetően három folthalmaz-transzformációt tesz lehetővé: mozgatást, összeolvasztást és eróziót, ezek menüpontok formájában elérhetőek a felhasználó számára. Mozgatás után az egyes foltok egymást átfedhetik, ezért a foltok között egy kirajzolási sorrend lesz definiálva. A program képes lesz bővítmények kezelésére olyan módon, hogy a fájlrendszerben a megfelelő mappába elhelyezett dll-eket indításkor detektálja és betölti, ezután az új dll által nyújtott plusz transzformációkat is elérhetővé teszi a felhasználói felületen.

Egy nyers kép betöltésekor az alkalmazás automatikusan felderíti a foltokat a kép pixeleinek átlátszósága alapján: a teljesen átlátszó pixelek nem lesznek folt részei, a többi közül pedig egybefüggő halmazok alkotnak egy-egy foltot.

Ami az alkalmazás megvalósítását illeti: Model-View-Viewmodel felépítésű lesz.

## Technikai paraméterek

A definiált alkalmazást .NET platformra, WPF felhasználásával C# nyelven készítjük el. Csak .NET keretrendszerrel rendelkező, WPF-et támogató - például Windows 7 és 8 operációs rendszeren - lesz futtatható. Az alkalmazásnak nem lesz különösebb hardverigénye. A fejlesztés és tesztelés az alábbi konfigurációjú számítógépen fog történni:

-Windows 8.1 x64 operációs rendszer

-Intel Core i5-2450M CPU @ 2.50 GHz

-6GB RAM

-ATI Mobility Radeon HD7650M 2GB videokártya

-1600X900 felbontású kijelző

## Szótár

**Pixel:** Színnel és koordinátákkal rendelkező képpont. Egy pixel pontosan egy foltnak a része.

**Szomszédos pixelek:** A pixel a közvetlenül fölötte, alatta, mellette lévő pixelekkel, vagyis összesen legfeljebb 4 másikkal lehet szomszédos.

**Nem teljesen körbevett pixel:** Olyan pixel, amelynek 4-nél kevesebb szomszédja van a foltban.

**Folt:** Egy vagy több pixelből álló, nem feltétlenül egybefüggő pixelhalmaz.

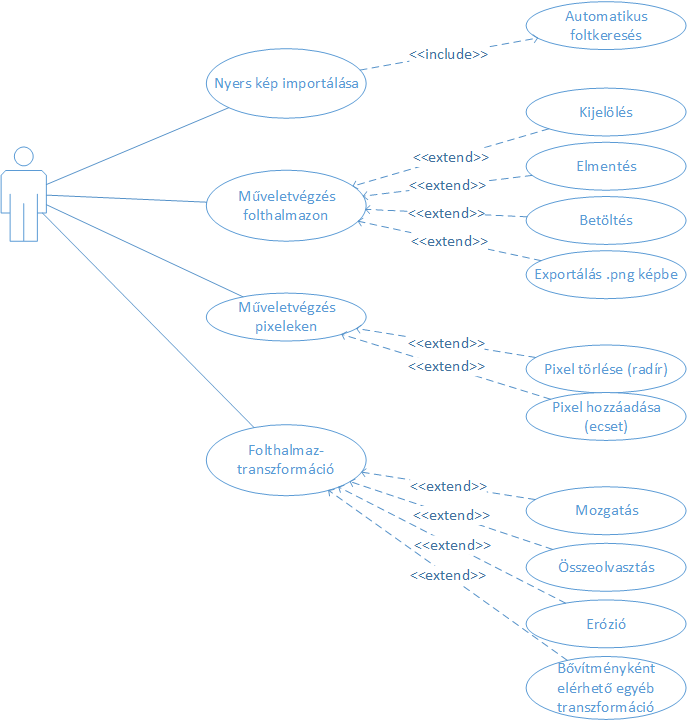
**Folthalmaz:** Több különböző folt együttese.

**Összeolvasztás**: Az éppen kijelölt folthalmazból egyetlen foltot előállító művelet, melynek elvégzése után az összes kiválasztott pixel egy foltot fog alkotni.

**Erózió**: Folthalmazon végezhető transzformáció, melynek során a nem teljesen körbevett pixelek eltávolításra kerülnek az egyes foltokból.

## Use-case diagram

A program az alábbi funkciókat támogatja:



# Tervezés

## Architektúra

## Modell

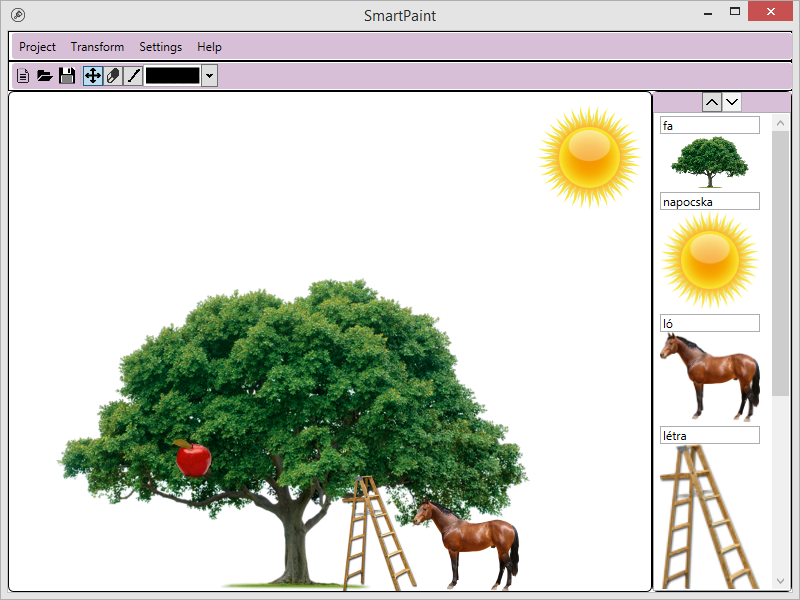
## UI terv

# Megvalósítás

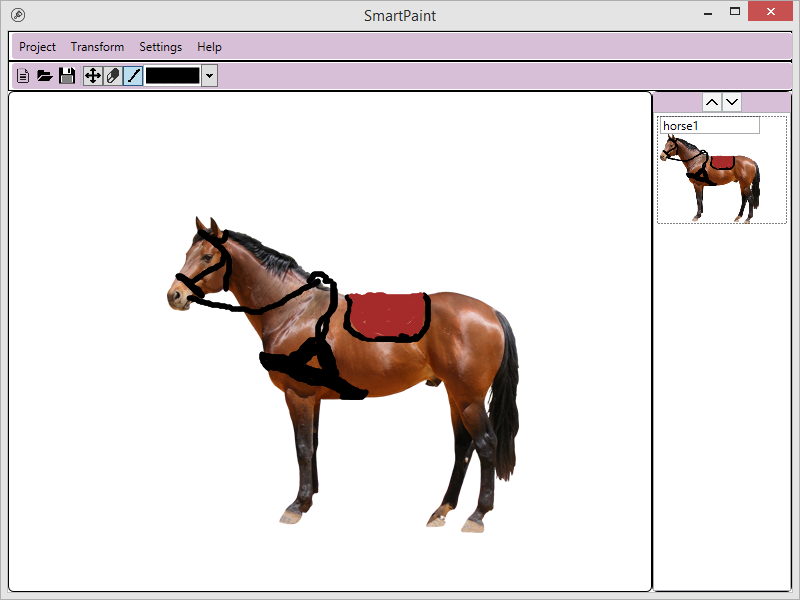
## Osztálydiagram

## Képernyőképek

Az alábbi képen látható, hogy a programba betöltött különböző alakzatok tetszőlegesen mozgathatóak a vásznon, és a kirajzolási sorrend is állítható a nyilak segítségével. A képeket természetesen az átlátszósági információ figyelembevételével rajzolja ki a program.



A beimportált képekre tetszőlegesen kiválasztott színnekkel rajzolhatunk:



# Összefoglalás

## Kihívások, tanulságok

A fejlesztés során mélyebben megismertük a .NET binding sokrétűségét, és a WPF felhasználói felület gazdag funkcionalitását. Ezen kívül az eróziót előállító képfeldolgozó algoritmust is megismertük.

## Továbbfejlesztési lehetőségek

A SmartPaintet természetesen rengeteg hasznos funkcióval lehetne még bővíteni, aminek csak a fejlesztőcsapat ideje szab határt. Néhány lehetőség a teljesség igénye nélkül:

* Több transzformáció
* Több támogatott nyelv
* Többféle méretű, alakú, illetve stílusú ecset és radír
* Átméretezhető rajzvászon
* Olyan felhasználói élményt javító funkciók, mint például visszavonás, kilépés előtt változások mentésének megkérdezése
* Professzionálisabb megjelenésű felhasználói felület, egyedi ablakkal és témával

# Telepítési útmutató

Mivel az alkalmazás hordozható (portable), nem szükséges telepíteni. Csak ki kell csomagolni a mellékelt .zip archívumot, és ezután a SmartPaint.exe alkalmazást már el is lehet indítani. A futtatáshoz szükséges minimális rendszerkövetelmények az alábbiak:

* Operációs rendszer: Microsoft Windows 8, vagy egyéb, .NET 4.0-t tartalmazó rendszer
* Legalább 10 MB szabad lemezterület
* Legalább 2 GB RAM memória
* 1 GHz-es, vagy jobb processzor

# Hivatkozások

A fejlesztés során felhasznált eszközök listája:

* Microsoft Visual Studio Ultimate 2012: A célunknak legjobban megfelelő fejlesztőeszköz.
* WPF Extended Toolkit: Egyedi WPF komponenseket tartalmazó eszköztár. A SmartPAint felhasználói felületén látható színválasztó komponenst innen vettük.
* DotNetZip: A program által használt .spt projektformátumba való mentéshez használtuk fel. A .spt tulajdonképpen egy becsomagolt .zip archívum, amit a DotNetZippel állítottunk elő.
* Git: A közös munkát nagymértékben megkönnyítette a Git verziókezelő rendszer.
* GitHub: A forráskód online hosztolásához a GitHubot választottuk.
* SourceTree: Grafikus Git felhasználói felület, melynek használata jóval kényelmesebb, mint a Git parancssoré.
* Microsoft Visio: Dokumentációs eszköz, amit az osztálydiagram és use-case diagram elkészítéséhez használtunk.