Vysoké učení technické v Brně
Fakulta informačních technologií

Tvorba uživatelských rozhraní – Fáze I

Aplikace DrawEasy

8. listopadu 2024

Kapitán: Patrik Uher
Alisher Mazhirinov
Václav Malina
Jakub Venera

Obsah

1 Úvodní strana

V rámci našeho projektu se s týmem zaměřujeme na vývoj aplikace pro kreslení. Každý z nás provedl výzkum existujících řešení a realizoval uživatelské průzkumy, abychom lépe porozuměli potřebám a preferencím našich cílových uživatelů. Tento přístup nám pomůže vytvořit efektivní a inovativní aplikaci.

2 Specifikace

2.1 Kdo je konkrétní uživatel

Student na vysoké nebo střední škole. Uživatel dbá na to, aby rozhraní bylo jednoduché, aby se co nejrychleji dostal do režimu, kdy může kreslit, a aby měla aplikace dobré výchozí nastavení.

2.2 Co potřebuje uživatel od aplikace

Uživatel potřebuje aplikaci na malování která se může použít jako substituce papíru, jako pomocný nástroj při učení (rozepsání příkladů, řešení problémů) a na vytváření rychlých náčrtů.

3 Uživatelský průzkum

3.1 Patrik Uher, xuherp02

Požadavky od uživatele:

- Uživatel přednostně potřebuje od aplikace způsob kreslení/psaní (tužka, štětec, psaní přes klávesnici), způsob mazání (guma, zvětšování velikosti gumy) a změnu barev (color wheel, předem definované výchozí barvy pro rychlý výběr).
- Uživatel preferuje rozhraní, které mu nebrání v malování tím, že by na začátku musel nastavovat velikost papíru, formát papíru apod. Upřednostňuje aplikace, které mají rychlé nasazení, aby je uživatel mohl používat téměř okamžitě.
- Uživatel oceňuje aplikace s jednoduchým rozhraním, aby si po delších přestávkách nemusel znovu osvojovat jejich ovládání. Proto je také důležité dodržovat běžné konvence, které používají ostatní populární aplikace na kreslení.

3.2 Alisher Mazhirinov, xmazhi00

Uživatelé v mém průzkumu si přejí mít jednoduché a přehledné rozhraní, aby mohli okamžitě začít pracovat. Ocenili by co nejjednodušší ovládání a také možnost mít v aplikaci průvodce, který by je provedl základními funkcemi.

Mezi dalšími přáními se objevily:

- Nápovědné bubliny (tooltips).
- Podpora pro pixel art.
- Přítomnost mnoha štětců, efektů a filtrů umožňuje realizovat kreativní nápady.

Naopak, uživatelé zmínili několik nevýhod stávajících aplikací:

- Ztráta kvality obrázku při změně velikosti, pokud dojde k narušení poměru stran.
- Rizika ztráty dat v důsledku nesprávného ukládání práce nebo selhání aplikace.
- Složitost rozhraní.

3.3 Václav Malina, xmalin33

Co přesně uživatele od aplikace potřebují:

- Přehledné a snadno použitelné rozhraní
- **Nástroje na kreslení**: Štětec, tužka, výplň barvou (kbelík), sprej. Uživatelé chtějí rychle kreslit jednoduché tvary, mít možnost přidávát text, jednoduchou úpravu chyb.
- Možnost rychle vybrat z různých přednastavených barev, případně míchat vlastní.
- Možnost načtení obrázkových souboru a export upravené verze souboru.

3.4 Jakub Venera, xvener01

Požadavky od uživatele:

- **Intuitivní nástroje a okamžitá použitelnost**: Nástroje na kreslení/úpravu, intuitivní změna barev, inteligentní výběr, možnost začít okamžitě používat aplikaci bez pro něj zbytečných nastavení, možnost načtení obrázků a jejich následná úprava, exportování upravených souborů.
- **Srozumitelné ikonky** : Jednoznačné ikonky, aby se co nejvíce zmenšila uživatelova učící křivka.
- **Podpora a přizpůsobení klávesových zkratek**
- **Dočasné ukládání historie úprav**

4 Průzkum existujících řešení

4.1 Patrik Uher, xuherp02

4.1.1 Rnote

Jednoduchá, open source aplikace na tvorbu vektorové grafiky.

Výhody:

- **Jednoduchý design**: Jednoduchý a lehce naučitelný design.
- **Rychlý přístup k nástrojům**: Rychlý přístup ke všem důležitým nástrojům.
- **Historie souborů**: Historie nedávno použitých a uložených souborů pro rychlý přístup.

Nevýhody:

- **Chybí pokročilé nástroje a funkce.**
- **Minimální podpora pro manipulaci rastrových obrázků.**
- **Žádná podpora vrstev.**

4.1.2 Pinta

Aplikace na tvorbu rastrové grafiky.

Výhody:

- **Osa historie úprav**: Historie provedených funkcí je představena jako osa, přes kterou se dá proklikávat.
- **Standardní menu nástrojů**: Menu pro výběr nástrojů a funkcí podporuje konvenci.
- **Dobrá podpora manipulaci obrázků.**

Nevýhody:

- **Neintuitivní pokročilé funkce**: Neintuitivní (staré) vyhledávání pokročilých funkcí.
- **Chybí šablony papíru**: Chybý výběr šablon pro standardizované velikosti papíru.
- **Nedostatek vysvětlivek**: Neexistující vysvětlivky u méně používaných funkcí.

4.2 Alisher Mazhirinov, xmazhi00

4.2.1 Aseprite

Populární aplikace pro tvorbu pixelové grafiky a animací.

Výhody:

- **Jednoduchost a minimalismus rozhraní**: Zjednodušené a intuitivní rozhraní umožňující uživatelům soustředit se na úkol.
- **Klávesové zkratky**: Možnost přizpůsobit klávesové zkratky zlepšuje uživatelský zážitek a urychluje práci.
- **Rychlost a nízké požadavky**: Program spotřebovává málo zdrojů, což je důležité pro rychlou práci při kreslení.

Nevýhody:

- **Úzká specializace**: Aseprite není vhodný pro vytváření složité nebo vysoce detailní grafiky. Je to čistě nástroj pro pixelovou grafiku.
- **Omezené editační nástroje**: Nemá tak silné funkce pro práci s barvami, filtry nebo složitými efekty jako sofistikovanější grafické editory.

4.2.2 GIMP

GNU Image Manipulation Program. Bezplatný a výkonný editor rastrové grafiky s mnoha funkcemi, často označován jako alternativa k `Photoshopu`.

Výhody:

- **Podpora mnoha formátů**: Podpora exportu i importu souborů v různých formátech, včetně PSD.
- **Široká škála funkcí**: GIMP nabízí pokročilé nástroje pro práci s obrázky, jako jsou vrstvy, masky, gradienty, filtry, skripty pro automatizaci a mnoho dalšího.

Nevýhody:

- **Složité rozhraní**: Rozhraní může být pro začínající uživatele přehlcené, zejména kvůli velkému množství funkcí. Program může být obtížně ovladatelný.
- **Náročnost na výkon**: GIMP může pracovat pomalu s velkými soubory nebo složitými projekty na slabších počítačích.

4.3 Václav Malina, xmalin33

4.3.1 KolourPaint

Jednoduchá aplikace pro kreslení a úpravu obrázků, která nabízí základní nástroje pro tvorbu a úpravu grafiky.

Výhody:

- **Rychlý přístup k funkcím**: Většina funkcí je přístupná během několika kliknutí, což zajišťuje rychlou a snadnou práci.
- **Efektivita díky klávesovým zkratkám**: Podpora klávesových zkratek pro rychlý výběr funkcí zlepšuje efektivitu.
- **Přizpůsobitelné rozhraní**: Uživatelé mohou přizpůsobit rozhraní podle svých potřeb, jednotlivé panely lze libovolně přemisťovat.

Nevýhody:

- **Omezená podpora touchpadu**: Chybí možnost přiblížení plátna pomocí touchpadu, což omezuje komfort při práci.
- **Nevyužitý prostor v rozhraní**: Velká část grafického rozhraní zůstává nevyužitá, což vede ke ztrátě místa, které by mohlo být věnováno zvětšení ostatních prvků rozhraní.
- **Omezené možnosti volby barev**: Aplikace nenabízí možnost výběru barev, což omezuje flexibilitu při kreslení.

4.3.2 Drawing

Jednoduchá grafická aplikace pro kreslení a úpravu obrázků v prostředí GNOME, poskytující základní nástroje pro tvorbu a úpravu grafiky s důrazem na minimalismus a snadné použití.

Výhody:

- **Efektivní uspořádání rozhraní**: Dobře využitý prostor pro rozhraní zajišťuje pohodlnou práci.
- **Rychlá volba barev**: Intuitivní výběr barev umožňuje rychlou a snadnou volbu palety.
- **Rozšířené možnosti díky filtrům**: Podpora filtrů poskytuje širší možnosti úprav obrázků.

Nevýhody:

- **Omezené funkce**: Aplikace nabízí příliš málo funkcí, což omezuje možnosti uživatele.
- **Duplicitní nástroje kreslení**: Funkce tužky a zvýrazňovače jsou téměř totožné, stačila by jedna s možností nastavení průhlednosti.

- **Chybějící možnost vložení z clipboardu**: Nelze vložit obrázek přes clipboard, což komplikuje práci s externími soubory.
- **Nedostatek klávesových zkratek**: Chybí podpora klávesových zkratek typu `Ctrl + "+"` a pro výběr všeho `(Ctrl + A)`.

4.4 Jakub Venera, xvener01

4.4.1 MyPaint

Open-source grafická aplikace, zaměřená především na digitální malbu a kreslení, s důrazem na jednoduchost a plynulost práce.

Výhody:

- **Minimalistické a přehledné rozhraní**.
- **Nekonečné plátno**: Nabízí neomezenou velikost pracovního prostoru.
- **Nástroje pro kreslení a malování s realistickým chováním štětců**.

Nevýhody:

- **Chybí pokročilé nástroje pro úpravu**.
- **Malá podpora formátů souborů**.

4.4.2 InkScape

Open-source editor pro vektorovou grafiku.

Výhody:

- **Široká škála nástrojů pro vektorovou grafiku**: Nabízí širokou škálu nástrojů pro tvorbu a úpravu vektorové grafiky, včetně práce s křivkami, tvary, textem a trasováním bitmap.
- **Podporuje mnoho formátů souborů**.
- **Možnost rozšíření a přizpůsobení rozhraní**: Uživatelé mohou přidávat rozšíření a přizpůsobit si rozhraní i funkce podle svých potřeb.

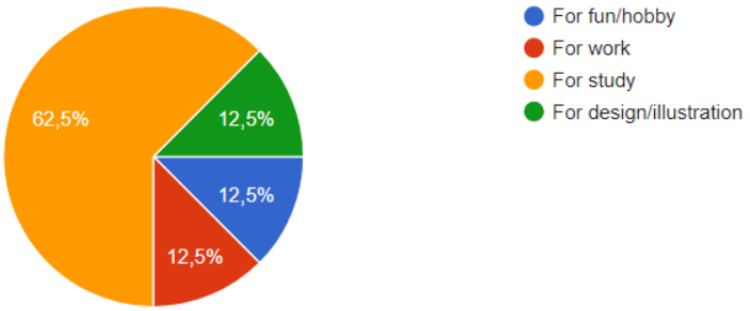
Nevýhody:

- **Delší učící křivka pro nové uživatele**: Pro nové uživatele může být rozhraní a práce s vektorovou grafikou komplikovaná.
- **Omezené pokročilé funkce**: Chybí některé pokročilé funkce a nástroje, které nabízí komerční software.
- **Omezená podpora pro barevné profily**.

5 Výsledky průzkumu

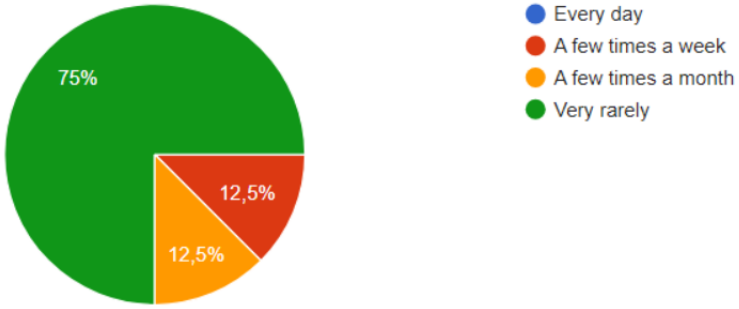
Během průzkumu jsme shromáždili data, která jsou jasně znázorněna pomocí tří diagramů.

K čemu nejčastěji používáte aplikace na kreslení?



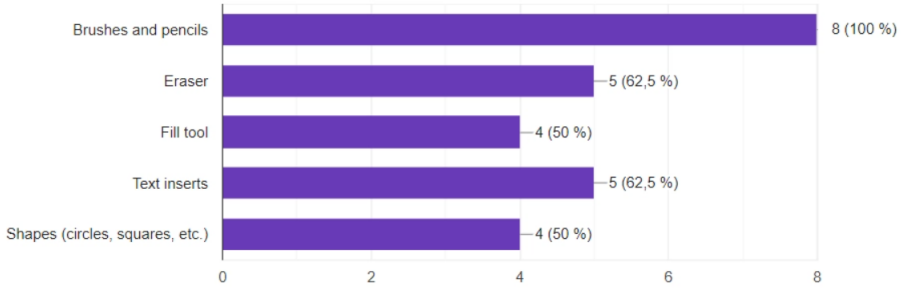
Obrázek 1: Pie chart 1

Jak často kreslíte na počítači?



Obrázek 2: Pie chart 2

Jaké kreslící nástroje jsou pro vás nezbytné?



Obrázek 3: Diagram 1

5.1 Co uživatelé potřebují

- **Nástroje pro kreslení**: Tužka, štětec, guma, výplň, vložení textu, tvary.
- **Barevná paleta**: Možnost rychlého výběru přednastavených barev a míchání vlastních barev.
- **Import/export souborů**: Načítání obrázků, jejich úprava a následný export do různých formátů.
- **Rychlé nasazení**: Okamžitý přístup k funkcím bez složitých přednastavení, např. velikosti nebo formátu papíru.
- **Minimalismus**: Uživatelé oceňují jednoduchá rozhraní, která minimalizují učící křivku.

5.2 Nevýhody existujících aplikací

- **Složitost rozhraní**: Některé aplikace jsou příliš komplikované nebo přehlcené funkcemi.
- **Nedostatečná optimalizace**: Některé aplikace jsou náročné na výkon nebo mají problémy s funkcemi jako je zoomování.
- **Omezené nástroje a funkce**: Některé aplikace postrádají pokročilé funkce nebo jsou zaměřené na specifické typy grafiky.

5.3 Inspirace pro vlastní návrh

- **Minimalismus a intuitivní rozhraní**: Z průzkumu víme, že klíčem k dobré zkušenosti je jednoduché a přehledné rozhraní. Naším cílem je udělat aplikaci, kde se uživatel rychle dostane k základním nástrojům a aby mohl začít kreslit během pár vteřin po spuštění, bez zbytečných komplikací.
- **Rychlé a jednoduché volby barev a nástrojů**: Inspirací je aplikace Drawing, která má snadný výběr barev a rychlý přístup k nástrojům. Chceme nabídnout přednastavené barvy a možnost míchat si vlastní přes jednoduchou paletu. Ikony nástrojů budou přehledné a srozumitelné, aby se v nich uživatelé snadno vyznali.
- **Podpora pro import/export souborů a dočasné ukládání**: Aplikace by měla podporovat import obrázků v různých formátech a snadný export upravených souborů. Přidáme také automatické ukládání, aby byla práce uživatelů chráněná a minimalizovalo se riziko ztráty dat.

6 Popis vývoje

Projekt je postaven na principu více frontendů (FE) pracujících na jednom backendu (BE). Pro frontend používáme technologie `GTK`, `PyGTK`, `PyQt` a `Qt`, které umožňují vytvořit různé rozhraní pro interakci s uživatelem. Všechny tyto rozhraní jsou propojeny s jediným backendem postaveným na knihovně `Cairo Graphics`.

`Cairo Graphics` zajišťuje veškeré grafické zpracování na straně backendu a poskytuje nástroje pro tvorbu a správu 2D grafických prvků, jako jsou čáry, tvary a text. Toto grafické jádro umožňuje vykreslování vizuálních prvků, které se zobrazují přes libovolné zvolené frontendové rozhraní.

6.1 Cairo API

`Cairo API` používá 4 hlavní koncepty ke kreslení:

- Source (zdroj) - barva nebo gradient, který se bude používat při kreslení.
- Destination (cílová plocha) - cílová plocha, na kterou se bude kreslit.
- Mask (maska) - mezistupeň mezi zdrojem a cílovou plochou, na kterém se provádějí změny, než jsou aplikovány na cílovou plochu.
- Path (cesta) - cesta, která je zapsaná na masce a později bude překreslena na cílovou plochu.

6.1.1 Datové struktury

Cairo Context:
Cairo Context je hlavní datová struktura, ve které jsou uloženy všechny části potřebné ke kreslení (zdroj, cílová plocha, maska a cesta). Každá část se může libovolně měnit pomocí dalších funkcí/metod.
Příklad: `cairo_t (C), Cairo::Context (C++), cairo.Context (Python)`

Uložené datové struktury v cairo_t:
`cairo_path_t, Cairo::Path, cairo.Path` - Cesta
`cairo_pattern_t, Cairo::Pattern, cairo.Pattern` - Zdroj
`cairo_surface_t, Cairo::Surface, cairo.Surface` - Cílová plocha
`cairo_matrix_t, Cairo::Matrix, cairo.Matrix` -Transformační Matice

6.1.2 Cairo funkce

Většina Cairo funkcí je naprogramována jako metody Cairo objektu (v C se předává do každé funkce ukazatel na cairo_t a každá funkce má prefix "cairo_").
Metody s x a y body mohou být buď absolutní, nebo relativní k aktuálnímu, ke kterému se přidá offset, s využitím metod s pozměněným jménem s "rel".
Ve výchozím nastavení Cairo smaže cestu poté, co se zapíše do cílové plochy, ale s využitím postfix "_preserve"se cesta nevymaže.

- Nastavení barvy zdroje.
  RGBA jsou reálná čísla od 0 do 1
  `set_source_rgb(red, green, blue, 1)`
  `set_source_rgba(red, green, blue, alpha)`
- Nastavení šířky čáry ke kreslení.
  `set_line_width(width)`
- Nastavení kde bude začínat kreslený objekt.
  `move_to(x, y)`
  `rel_move_to(x, y)`
- Nakreslení cesty čáry do masky.
  `line_to(x, y)`
  `rel_line_to(x, y)`
- Nakreslení masky do cílové plochy.
  `stroke()`
  `stroke_preserve()`

- Nakreslení cesty obdélníku do masky.
  `rectangle(x, y, width, height)`
- Vyplnění objektu ohraničeným cestou v masce do cílové plochy.
  `fill()`
  `fill_preserve()`
- Nakreslení cesty Bézierové křivky do masky.
  `curve_to(xc1, yc1, xc2, yc2, destx, desty)`
  `rel_curve_to(xc1, yc1, xc2, yc2, destx, desty)`
- Uzavření nedokončené cesty v masce nejkratší cestou.
  `close_path()`
- Změna transformační matice pro násobení bodů x a y přes faktor sx a sy.
  `scale(sx, sy)`
- Přidání transformační matice k aktuální transformační matici.
  `transform(transformation_matrix)`

Metody uvedené zde představují ty nejpoužívanější a nejzákladnější. Rozsáhlá dokumentace je dostupná na stránkách Cairo Graphics[1].

6.2 Napojení GTK na Cairo

6.2.1 Vytvoření a nastavení oblasti pro vykreslování:

- GTK vytvoří okno a widget `Gtk.DrawingArea`, který slouží jako plátno pro vykreslování.
- Tento widget může zpracovávat události a také iniciovat vykreslování pomocí signálu `"draw"`.

6.2.2 Inicializace kontextu vykreslování:

- Při přijetí signálu `"draw"` GTK automaticky vytvoří objekt `cairo.Context` — kontext pro vykreslování.
- Tento objekt spravuje grafické parametry jako je barva, tloušťka čar, transformace a také samotný proces vykreslování.

6.2.3 Zpracování událostí a vykreslování:

- GTK zavolá obslužnou funkci pro signál `"draw"`, do které předá objekt `cairo.Context`.
- Uvnitř obslužné funkce se používá API Cairo pro vykreslování prvků rozhraní na plátno.

6.2.4 Interakce s uživatelem:

- GTK spravuje interakci s uživatelem, zpracováváním událostí, jako jsou kliknutí myší, pohyby myši, stisky kláves a další.
- Tyto události mohou vést ke změně dat, která budou později vykreslena pomocí Cairo, nebo ke změně stavu rozhraní.

[1] Viz `https://www.cairographics.org/manual/`

6.2.5 Aktualizace a opětovné vykreslení:

- Po změně dat nebo stavu rozhraní (např. změna barvy nebo tvaru) se zavolá metoda `queue_draw` na widgetu.
- Tato metoda spustí nový signál `"draw"`, což vede k opětovnému zavolání obslužné funkce pro vykreslování a aktualizaci obrazu na plátně.

6.3 Napojení Qt na Cairo

V `Qt` jsou všechny interakce uživatele (kliknutí myší, stisk kláves) reprezentovány jako události, které obsahují potřebné informace o akci. Například při kliknutí myší vznikne `QMouseEvent`, který `Qt` pošle příslušnému widgetu. Klávesové události fungují podobně s objekty `QKeyEvent`.

6.3.1 Hlavní události

- `MousePressEvent` a `MouseReleaseEvent` — události pro stisk a uvolnění tlačítka myši.
- `KeyPressEvent` a `KeyReleaseEvent` — události pro stisk a uvolnění klávesy.

6.3.2 Signály a sloty

Qt využívá signál-slot systém, kde widget při určité akci vysílá signál (např. `clicked()` při kliknutí na tlačítko), který lze propojit s metodou (slotem) pro provedení určitého úkolu.

6.3.3 Kreslení a widgety

Qt používá malovací systém založený na třídě `QPainter` pro vykreslení všech prvků (widgetů) na obrazovce. Pro vlastní grafiku lze přepsat metodu `paintEvent()` a vykreslovat pomocí `QPainter`, přičemž kreslení obrazů se provádí metodou `drawImage` na objektu `QImage`.

6.3.4 Pycairo a QImage v backendu

Pycairo vytvoří kreslitelné plátno pomocí `ImageSurface` a vykreslí grafiku, kterou pak předá objektu `QImage` v PyQt (ve formátu `QImage.Format_ARGB32`), kde se zobrazí pomocí `QPainter`.

6.4 Patrik Uher, xuherp02

6.4.1 GTKmm

`GTKmm` je C++ binding na C knihovnu GTK. Využívá objektový přístup a signál-slot mechanismus, což usnadňuje práci s rozhraním a zpracování uživatelských akcí.

Zdůvodnění výběru:
GTK je sada nástrojů pro tvorbu GUI aplikací. GTK je moderní a velmi používaný toolkit, pro tvorbu GUI aplikací na linuxu je to velmi dobrá volba.

`Cairomm` je C++ binding na C knihovnu Cairo. Cairo je knihovna na kreslení vektorové a rastrové grafiky. Výběr této technologie pro aplikaci byl jednoduchý, protože hodně konkurenčních aplikací taky používá cairo.

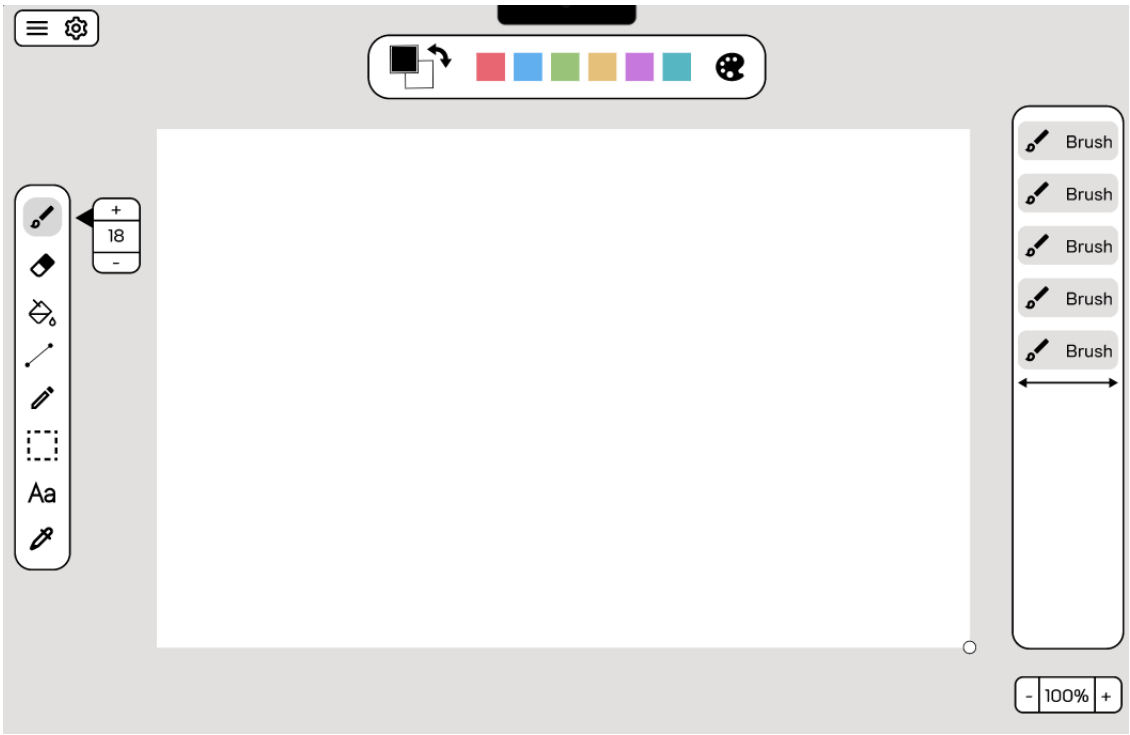
Logická implikace navrženého GUI:
Návrh je minimalistický, intuitivní a moderní. V hlavních částech, jako jsou nástroje na kreslení a plocha na kreslení, se návrh příliš neliší od konkurenčních aplikací, aby se uživatel nemusel učit nové

uspořádání a naopak přesně věděl, kde je vše, co potřebuje. Na druhou stranu je návrh odlišný, protože využívá prvky moderních aplikací.

Návrh je koncipovaný tak, že až na pár funkcí se ke každé části programu uživatel dostane maximálně na dvě kliknutí myší, a někdy dokonce jen pomocí krátké klávesové zkratky. Tato část návrhu je velmi důležitá, aby si uživatel nemusel pamatovat složité postupy pro dosažení požadovaných funkcí.

Na levé straně okna jsou nástroje na kreslení. Nástroje jsou seskupeny podle jejich důležitosti a tak, jak podobné nástroje seskupují konkurenční aplikace. Po kliknutí na nástroj se zobrazí nastavení specifická pro daný nástroj. V levém horním rohu je menu a nastavení, což je standardní umístění pro moderní GUI aplikace. Uprostřed nahoře je plocha pro výběr barev, která je vždy zobrazena, aby uživatel mohl kdykoli měnit barvu. Toto řešení je důležité, protože průzkum ukázal, že měnění barev je pro uživatele podstatné. Na výběr je šest barev, což je standardní počet pro základní barevné palety. V pravé části aplikace je osa historie, kterou může uživatel procházet klikáním nebo ji jednoduše přetahovat. Zbytek plochy je určený na kreslení, aby uživatel mohl co nejvíce využít plochu aplikace ke své tvorbě.

6.4.2 Navrh GUI



Obrázek 4: GUI xuherp02

6.5 Alisher Mazhirinov, xmazhi00

6.5.1 PyGTK

`PyGTK` je Python wrapper pro knihovnu `GTK`, určenou k vytváření okenních aplikací s rozhraními, které podporují widgety, jako jsou tlačítka, vstupní pole a nabídky. Interakce s `GTK` se provádí prostřednictvím `GObject Introspection (gi)`, což umožňuje dynamicky používat funkce knihovny `GTK` v Pythonu. To výrazně zjednodušuje práci s knihovnou a umožňuje snadné vytváření a nastavení prvků rozhraní v Pythonu.

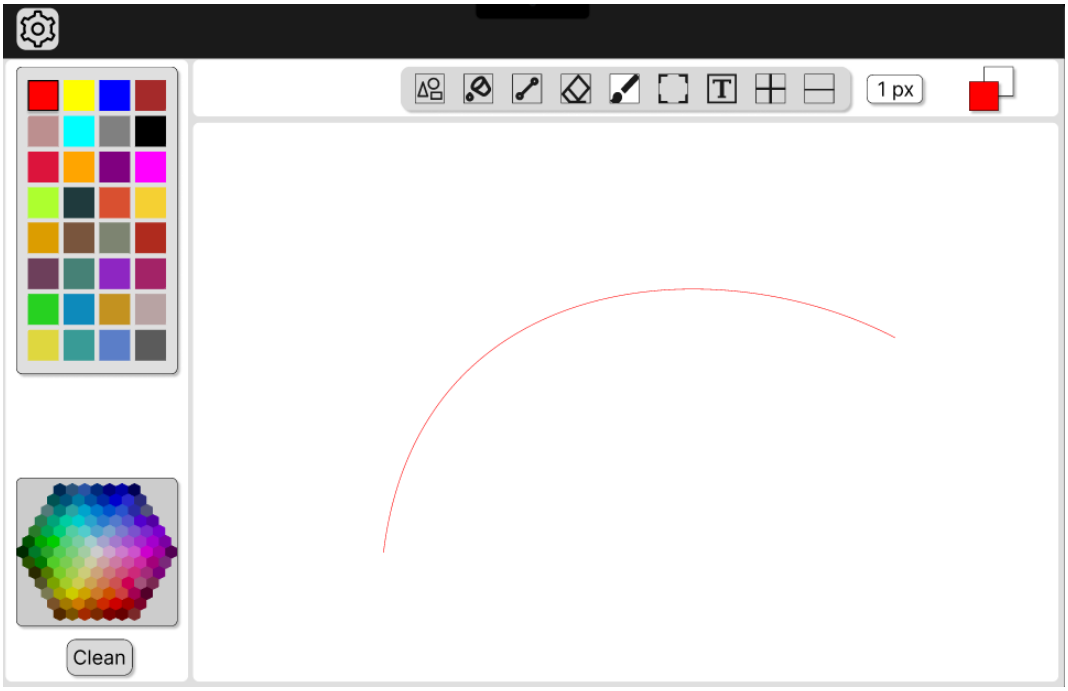
Zdůvodnění výběru:
`PyGTK` poskytuje vysoce úrovňové rozhraní pro práci s `GTK`, což umožňuje vývojářům snadno vytvářet grafické rozhraní. Jeho syntax a struktura jsou intuitivně srozumitelné. `GTK` nabízí bohatou sadu připravených widgetů, jako jsou tlačítka, textová pole, menu a další prvky rozhraní, které urychlují vývoj, a tudíž není potřeba vytvářet vše od začátku.

Logická implikace navrženého GUI:
Na základě provedeného výzkumu jsem identifikoval klíčové body, které jsem implementoval do své aplikace. Uživatelé chtějí začít s aplikací co nejrychleji a neztrácet čas zaškolováním, takže rozhraní by mělo být co nejjednodušší a intuitivní.

Všechny základní funkce a nástroje – štětce, barvy, režimy kreslení – jsou proto umístěny na jednom místě, což umožňuje snadné a rychlé přepínání mezi nimi a šetří uživatelům čas. Důraz je kladen na srozumitelné ikony a ovládací prvky, které zajistí, že aplikaci může okamžitě využívat i nový uživatel.

Kromě toho existuje možnost exportu práce do různých formátů.Uživatel to může provést v levém rohu aplikace po stisknutí ozubeného kolečka a výběru potřebné funkce v menu.

6.5.2 Navrh GUI



Obrázek 5: GUI xmazhi00

6.6 Vaclav Malina, xmalin33

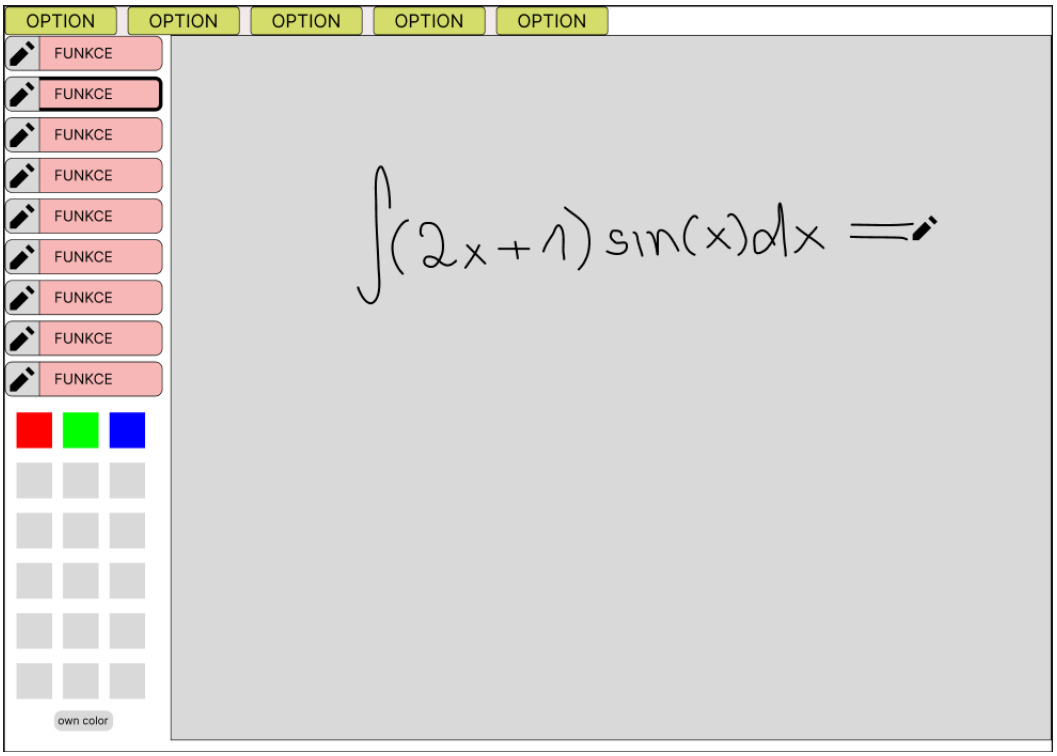
6.6.1 PyQt

`PyQt` je sada pythonových vazeb pro nástroje `Qt`, což je výkonný a komplexní framework pro vytváření grafických uživatelských rozhraní. Qt samo o sobě je napsáno v C++, ale `PyQt` poskytuje rozhraní, které umožňuje programátorům používat `Qt` API přímo v Pythonu. Díky tomu je `PyQt` velmi populární pro vývoj aplikací s moderním a responzivním rozhraním.

Zdůvodnění výběru:
`PyQt` jsem zvolit jako FE technologii, protože poskytuje spolehlivý a flexibilní framework, který podporuje profesionální vývoj GUI aplikací s intuitivním a moderním uživatelským rozhraním. `PyQt` je multiplatformní. To zajišťuje, že aplikace bude fungovat na různých operačních systemech.

Logická implikace navrženého GUI:
Nástroje pro kreslení jsou uživateli dostupné v levé horní části okna prostřednictvím tlačítek. Pro barvy má uživatel předpřipravená tlačítka v levé dolní časti okna, kde si kliknutim na příslušné tlačítko může vybrat vlastní barvu. Import a export obrázkových souborů je možný přes tlačítka option. Uživatelské rozhraní je minimální a uživatel má možnost začíst kreslit téměř ihned po spuštění aplikace.

6.6.2 Navrh GUI



Obrázek 6: GUI xmalin33

6.7 Jakub Venera, xvener01

6.7.1 Qt

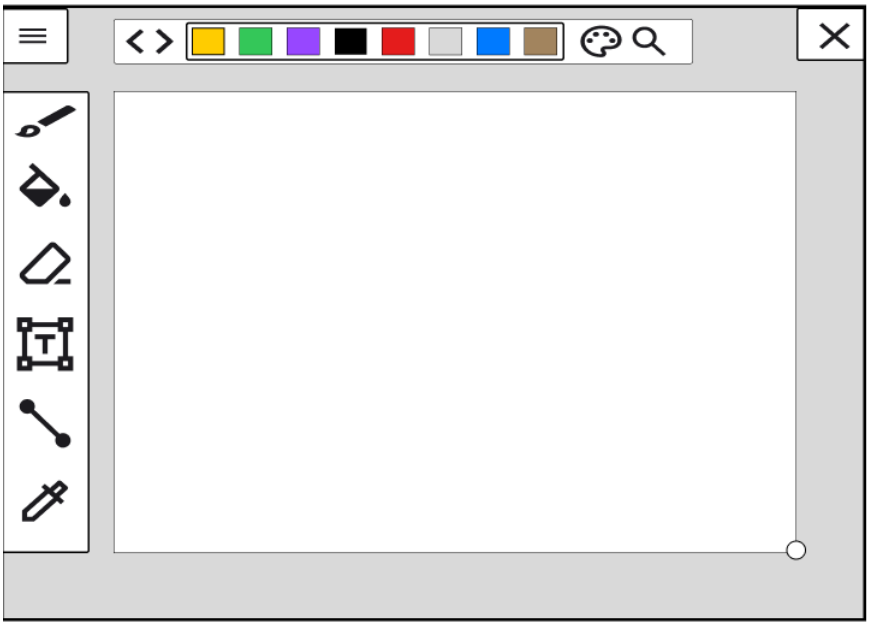
Multiplatformní framework pro tvorbu aplikací s grafickým uživatelským rozhraním, ale také pro vývoj bezobslužných aplikací a nástrojů. Je napsaný v C++, přičemž podporuje i mnoho dalších jazyků díky bindingům. Qt obsahuje širokou škálu modulů, které umožňují vytvářet aplikace pro různé platformy.

Zdůvodnění výběru:
Vybral jsem si `Qt`, protože je probíráno v rámci přednášky, umožňuje vyvíjet aplikace, které mohou běžet na různých operačních systémech a podporuje moderní C++ standardy a technologie.

Logická implikace navrženého GUI:

- Jasné, minimalistické prvky, které nezahlcují uživatele informacemi a umožňují mu rychle se zorientovat.
- Základní nástroje jsou umístěny na viditelných a snadno přístupných místech, jako je panel nástrojů umístěný na bočním panelu, což umožňuje uživateli okamžitě začít kreslit.
- Jednoduchá paleta barev, která obsahuje přednastavené barvy a také možnost míchat vlastní barvy. Je umístěna v horním panelu, aby byla uživateli snadno dostupná.
- Ikony pro různé nástroje jsou zvoleny tak, aby byly srozumitelné a snadno rozpoznatelné, což usnadňuje jejich používání.

6.7.2 Navrh GUI



Obrázek 7: GUI xvener01