GSoC 2024 Proposal

For

Sugarlabs

Idea: GTK4 Exploration

Table of Contents

Personal Details

Full Name: Ahmed Fatthi Ali Muhammed

Email: ahmedfatthi1040@hotmail.com

Another Email: ahmedfathy1060@gmail.com

GitHub Username: AhmedFatthy1040

LinkedIn: https://www.linkedin.com/in/ahmedfatthi1040/

Languages: Arabic (Native), and English.

Location: Cairo, Egypt.

Timezone: EET (Eastern European Time) UTC/GMT +2 hours

Pull Requests: (Merged Manually)
https://github.com/llaske/sugarizer/pull/1512

Note: Although the pull request appears as closed, it was actually merged manually because I made a slight mistake while pushing it. I inadvertently pushed the changes to the wrong branch.

In this pull request, I've addressed an issue where players could exploit a bug in the game. Essentially, they could cheat by hitting the pause button while the ball was mid-bounce, causing the game not to register the round, giving them an unfair advantage.

To fix this, I've implemented a solution that disables the pause functionality when the ball is in play after the first round. This ensures that players cannot pause the game during crucial moments, preserving the fairness and integrity of gameplay.

Summary

As a third-year computer science student at Helwan University, I've accumulated valuable experience in backend development and participated in three enriching internships across various domains within computer science.

During my first internship, I immersed myself in SOC Analysis, where I deepened my understanding of Linux fundamentals and honed my skills in Ubuntu. In the subsequent internship, I focused on refining my problem-solving abilities through competitive programming challenges. Most recently, I delved into backend development during my third internship, allowing me to further cultivate my expertise in this field.

In addition to my internship experiences, I've actively contributed to numerous projects, gaining proficiency across various technology stacks. My exposure to diverse projects has endowed me with the capacity to learn rapidly, making me a quick and efficient learner. Furthermore, my adeptness at self-learning has enabled me to adapt seamlessly to new environments and conquer challenges with ease.

Moreover, my college studies have provided me with a solid foundation in the C programming language. I've implemented essential data structures such as stacks, queues, and linked lists using C. Additionally, I've developed a C library encompassing various mathematical functions, further augmenting my skills and proficiency in the language.

GTK4 Exploration Project Details

Objective:

The objective of the GTK4 Exploration project is to migrate the Sugar learning platform from GTK3 to GTK4. This migration aims to ensure continued compatibility and support for Sugar as GTK3 approaches its end-of-life (EOL). By porting Sugar and its core activities to GTK4, the project seeks to facilitate the adoption of the latest GTK version, enabling users to benefit from enhanced features, improved performance, and long-term sustainability.

Impact:

1. **Enhanced Compatibility:** Migrating Sugar to GTK4 ensures compatibility with the latest GTK version, enabling users to leverage new features and enhancements introduced in GTK4. This compatibility ensures a smoother user experience and facilitates the adoption of future GTK updates.

2. **Improved Performance:** GTK4 offers optimizations and performance improvements over GTK3. By porting Sugar to GTK4, the project enhances the platform's performance, resulting in faster response times, reduced resource utilization, and improved overall efficiency.

3. **Extended Support:** As GTK3 approaches its EOL, maintaining compatibility with GTK4 ensures continued support and maintenance for Sugar. This extended support lifecycle ensures that users can rely on Sugar for their educational needs without worrying about obsolete technology or compatibility issues.

Required Technologies:

1. **C Programming Language:** Proficiency in C is essential for the GTK4 Exploration project, as it forms the foundation of GTK development. A strong understanding of C programming principles, syntax, and best practices is necessary for implementing the necessary changes and adaptations to migrate Sugar and its core activities to GTK4. With prior experience gained from projects undertaken during college, i already possess a solid foundation in C programming, which will be invaluable for this project.

2. **Python:** While the project primarily involves working with C for GTK development, familiarity with Python is beneficial, as Sugar also utilizes Python for certain components and scripting tasks. My existing experience in Python will enable me to work efficiently with Sugar's codebase and integrate Python-based functionalities as needed during the migration process.

3. **GTK (GIMP Toolkit):** Although I am currently learning GTK, my existing experience in C programming will provide a solid foundation for grasping GTK concepts and APIs quickly. Familiarity with GTK's architecture, APIs, and development paradigms is crucial for effectively migrating Sugar's graphical user interface components and ensuring compatibility with GTK4.

What’s new in GTK4 ?

GTK4 introduces several new features and improvements compared to GTK3, which can have a significant impact on Sugar and its development environment

- **Improved Performance:** GTK4 offers optimizations and enhancements that improve rendering performance, responsiveness, and overall efficiency. This can lead to a smoother and more responsive user experience in Sugar applications.

- **Modernized API:** GTK4 introduces a modernized API with cleaner, more consistent naming conventions and improved documentation. This makes it easier for developers to work with GTK4 and reduces the learning curve for newcomers to the framework.

- **Enhanced Hardware Acceleration:** GTK4 leverages hardware acceleration more effectively, resulting in better utilization of GPU resources and improved rendering performance. This can benefit Sugar applications, especially those with graphics-intensive components.

- **CSS-Based Styling:** GTK4 adopts a CSS-based styling system for theming and customization, providing more flexibility and control over the visual appearance of applications. This allows Sugar developers to create modern and visually appealing user interfaces with ease.

- **Redesigned Input Handling:** GTK4 introduces changes to input handling, including improvements in event handling and management. This can lead to more robust and reliable input handling in Sugar applications.

GNOME Recommended Approach for Migration

GNOME recommends a systematic approach to migrating from GTK 3.x to GTK 4, emphasizing thorough preparation and consideration of various changes and adaptations required during the transition. The recommended steps include

1. Preparation in GTK 3.x:
    - Avoid using deprecated symbols.
    - Enable diagnostic warnings.
    - Review and avoid using GTK-specific command line arguments.
    - Refrain from using widget style properties.
    - Review window creation flags.
    - Cease direct access to GdkEvent structs.
    - Avoid non-RGBA visuals.
    - Update usage of GtkBox padding, fill, and expand child properties.
    - Adjust GtkStyleContext getters usage.
    - Replace certain deprecated functions such as gdk_pointer_warp() and gdk_pixbuf_get_from_window().
    - Set a proper application ID.
    - Transition from gtk_main() and related APIs to new methods.
    - Minimize usage of gtk_widget_destroy().

2. Changes that need to be done at the time of the switch:
    - Adapt to various API changes including those related to GdkScreen, GdkVisual, GdkDeviceManager, GdkWindow, GdkEvent, grabs, coordinate, and GdkKeymap.
    - Convert UI files and adapt to changes in GtkBuilder API.
    - Update focus handling, keyboard shortcuts, and event controller APIs.
    - Adjust UI elements such as GtkButtonBox, GtkContainer, GtkScrolledWindow, and GtkHeaderBar to API changes.
    - Use GtkFixed instead of GtkLayout.
    - Update CSS styling and adapt to changes in drawing model.

3. Changes to consider after the switch:
    - Port to new list widgets.
    - Update usage of GtkFileChooser, GtkToolbar, GtkPopover, GtkMenu, GtkMenuBar, GtkMenuItem, GtkAspectFrame, GtkScale, GtkFileChooserButton, GtkBuildable, GtkAboutDialog, and GtkTreeView among others.
    - Adopt new Drag-and-Drop API.
    - Update to GtkFileChooser API changes.
    - Consider removing usage of blocking dialog functions and GtkBuildable API.
    - Adapt to GtkTreeView and GtkIconView tooltip context changes.
    - Adjust GtkSettings properties.

Migration Plan and Strategy

The migration plan for the GTK4 Exploration project involves a systematic approach to porting Sugar from GTK3 to GTK4 while ensuring minimal disruption to functionality and compatibility. The following steps outline the migration strategy:

- **Initial Assessment:** I will begin by conducting a comprehensive assessment of the existing Sugar codebase and dependencies to identify components that need to be migrated from GTK3 to GTK4. This assessment will help prioritize migration tasks and allocate resources effectively.

- **Setup Development Environment:** Set up a development environment with the necessary tools and libraries required for building and testing Sugar with GTK4. Ensure that the environment is configured to replicate the production environment as closely as possible to mitigate compatibility issues.

- **Component Migration:** Start by migrating minimal sugar-toolkit-gtk3 components necessary to support a basic "Hello World" activity. Focus on adapting activity and graphics classes to work with GTK4 while maintaining compatibility with existing functionality.

- **Activity Migration:** Proceed to migrate the "Hello World" activity itself to GTK4. Address any compatibility issues or bugs that arise during the migration process, ensuring that the activity functions correctly in the GTK4 environment.

- **Toolkit Component Migration:** Migrate the remaining toolkit components essential for Sugar to fully support GTK4. Prioritize components based on their importance to Sugar's functionality and test each migrated component rigorously to ensure compatibility and stability.

- **Activity Extension:** Extend the migrated "Hello World" activity to utilize the remaining toolkit components and rename it as a "Toolkit Test" activity. Test the extended activity to verify proper integration with all toolkit components and functionality.

- **Framework Migration:** Migrate the Sugar framework itself to support GTK4. This involves updating core modules, libraries, and dependencies to ensure full compatibility with GTK4. Test the migrated framework thoroughly to ensure stability and functionality.

- **Fructose Activity Migration (Optional):** If time permits, migrate the Fructose activity set, which is a set of core activities in Sugar, to GTK4. Prioritize activities based on their importance and usage within the Sugar ecosystem.

- **Documentation:** Throughout the migration process, maintain comprehensive documentation detailing the migration strategy, process, and any challenges encountered. Document best practices, tips, and troubleshooting techniques to facilitate knowledge sharing and future development efforts.

Timeline

Week 1-2: Planning and Setup

- Familiarize myself with the existing Sugar codebase.
- Set up the development environment for building and testing Sugar with GTK4.
- Begin drafting documentation for the migration process.

Week 3-4: Minimal Component Migration

- Start migrating minimal sugar-toolkit-gtk3 components necessary to support a basic "Hello World" activity.
- Focus on adapting activity and graphics classes to work with GTK4.
- Test the migrated components to ensure they function correctly in the GTK4 environment.
- Document the migration process and any challenges encountered.

Week 5-6: Hello World Activity Migration

- Continue the migration process by porting the "Hello World" activity itself to GTK4.
- Address any compatibility issues or bugs that arise during the migration.
- Test the migrated activity thoroughly to verify its functionality in the GTK4 environment.
- Update documentation with insights and solutions discovered during the activity migration.

Week 7-8: Toolkit Component Migration

- Proceed with migrating the remaining toolkit components required for Sugar to fully support GTK4.
- Prioritize components based on their importance to Sugar's functionality.
- Test each migrated component rigorously to ensure compatibility and stability.
- Update documentation with migration instructions for each toolkit component.

Week 9-10: Extending Hello World Activity

- Extend the migrated "Hello World" activity to utilize the remaining toolkit components.
- Rename the activity as a "Toolkit Test" to reflect its expanded functionality.
- Test the extended activity to verify that it properly integrates with all toolkit components.
- Document the process of extending the activity and any challenges encountered.

Week 11-12: Sugar and Fructose Migration

- Begin migrating the Sugar framework itself to support GTK4.
- Prioritize critical modules and functionalities within Sugar.
- Test the migrated Sugar framework to ensure it remains stable and functional.
- If time permits, start migrating the Fructose activity set to GTK4.

Week 13: Final Testing and Documentation

- Conduct comprehensive testing of the entire Sugar environment with GTK4.
- Identify and address any remaining issues or bugs.
- Finalize documentation for the migration process, including detailed instructions and troubleshooting tips.
- Prepare a final report summarizing the project's achievements, challenges, and contributions to the Sugar community.

Week 14: Buffer Week

- I will use this week as a buffer to address any unforeseen issues or complete pending tasks.
- Conduct final reviews of documentation and codebase.
- Ensure all project deliverables are met before the final submission deadline.