DSFML Additions and Improvements

Background

Bindings to existing code is important to the D programming language. Out of all the numerous things that D *does* have going for it, unfortunately man power is not one of them. DSFML is a binding and wrapper to SFML, a C++ multimedia library mostly aimed at video games. Due to using SFML as a base, DSFML has the best support on all desktop platforms and the most features out of all the current D multimedia and game development libraries. I have been working on DSFML since 2013 and it continues to evolve and become better.

The DSFML project itself composes of two distinct sub-projects: the back end which consists of a C API wrapper around SFML referred to as DSFMLC, and the front end which is the D API wrapped around DSFMLC, which is the actual DSFML library.

Project Synopsis

I will continue work on DSFML to not only improve its code base, but to also add relevant features to using D on more platforms. The main proposed improvement is to add support for iOS and Android to the library. Currently, SFML is able to target both iOS and Android operating systems and thanks to recent efforts with the LDC compiler, DSFML can be made to target those operating systems as well. In order to do this, the version of SFML that DSFMLC wraps will need to be upgraded, and the API will need to be modified in the back end and front end accordingly. I would also like to work on several bugs in the library to make sure that all features in DSFML work the same as they do in SFML. D deserves a multimedia library of the same caliber of SFML proper.

Given extra time, there are other things that could and should be improved. This includes tutorials, example code, tests, documentation, and the over all presentation (quality of the website, ease of getting started, presentation of documentation, etc.)

What This Brings to D

By including iOS and Android support to the library, this will help to increase awareness to D and show that D is a strong candidate for multimedia applications not only on desktop but also mobile.

Success Criteria

1. The addition of support for mobile platforms is the highest priority.

2. Improvements to the build systems to support the new targets

3. API updates to match the most recent SFML version.

4. Bug fixes for all improperly functioning features.

Optional work given enough time:

- Improvements to tutorials
- Improvements to the testing
- Add example code that can be built and run
- improvements to documentation
- improvements to the general api

Roadmap

- Upgrade the internally used SFML to the most recent version (2.3)
- Work out any bugs in getting the back end when building DSFMLC for iOS and Andoid
- Leverage the current work on LDC to link and build executables that can run on iOS and Android mobile devices
- Add testing and CI for the newly introduced iOS and Android targets
- Update the API to reflect the recent changes in SFML
- Work on bugs and issues reported by users
- Continue working on optional improvements as noted above

About me

I am a 27 year old computer science student who has been programming off and on for around 10 years, but only recently taken it seriously. I was just accepted to the University of Washington's CS program in Bothell, Washington and will start in the fall. DSFML was my first project when learning D, and I am excited to see it used.