**“Friendlier” – Experience Development Environment**

**Overview and Release Notes Version 1.1**

**November 2012**

# Synopsis

Friendlier is an Experience Development Environment created and developed by Xyglo. Friendlier allows you to easily create compelling user experiences - be they interactive (games, applications) or passive (presentations, demos). Friendlier provides a file editor and directory and system visualisation that allow you to interact with development tools on your system. This world allows interaction with 3D objects and provides REPL like behaviour for certain graphical elements defined in Xyglo’s *Brazil Framework*.

*Brazil* is the technology layer which underpins Friendlier and allows its flexibility. Brazil is a cross platform 3D component framework and included in this document is an overview of the Brazil Framework and a short introduction to creating applications in Brazil for C#/XNA for Windows Desktop and XBox, for MonoGame and natively for Android.

This document provides information explaining more about what Friendlier is, how it works, how you can get it and how you can use it to develop your own stunning experiences.

# Introduction to Friendlier and Brazil

At a basic level Friendlier is a text editor implemented in 3D. Why 3D? Well, because 3D is cool and as everyone knows [text editing is boring](http://www.sublimetext.com/blog/articles/it-pans-it-zooms-it). Friendlier is based around the standard model of text editors and borrowing many features common to things like vi, emacs, notepad, word etc. Files can be loaded, saved and edited. Editing takes place at a cursor. The cursor can be moved using the cursor keys on the keyboard or the mouse. New text can be inserted or deleted at the cursor position, you can create highlights, cut and paste.

Text Editing *is* boring though – and this isn’t fundamentally what Friendlier is about. We’re about experiences – having them, creating them, and Friendlier can help you do that. If you find that the text editor isn’t cutting it then let us know and we’ll try to improve it, however the focus of this project is on the creating a unique overall experience and not replacing what you have already. We don’t want to replace your IDE, your programming language or your text editor – we want Friendlier to be another way you can create cool things and we believe that Friendlier in combination with the Brazil framework is a compelling toolset.

## History

So, it’s the end of 2012 why did Xyglo end up with a kooky 3D text editor and touting its own framework? Well this is my approach to handling where we are in proprietary 3D and interactive consumer software technology. Without wanting to totally submit to the constant march of progress this is my attempt to make sense of the mess of 3D technology and new interfaces I see around me.

To whit:

There are a lot of target platforms out there which have accessible APIs and are relevant for software development. Windows, Mac, Android, IOS, Linux etc.

The APIs themselves move quickly and can be discarded as quickly as they appear

The underlying technology moves a lot more slowly and this is because it’s related to the hardware (OpenGL, DirectX, Mouse, Keyboard, Kinect, Touch etc)

# Using Friendlier

Friendlier is a text editor and project management interface which allows you to write, edit and compile code using your favourite language and compiler. It’s configurable – so Friendlier is something like a light IDE (Integrated Development Environment) for your favourite compiled programming language. This section introduces you to the main features of Friendlier and gets you up and running with the basics.

## Text Editing in Friendlier

Editing takes place by typing new characters, deleting characters and replacing characters in files. Use the SHIFT key in combination with the cursor keys to sweep out an area to be deleted or replaced. You can use the CONTROL key to skip words forwards or backwards.

The standard BufferView (a view on a file) fixes the character width to 80 columns and the buffer height to 40 rows. Files with lines longer than 80 columns will be indicated with a [>] mark outside the editing pane. By moving the cursor key right along this line (or by using the END key) the user can jump to the end of these lines and the editing pane will move accordingly.

Friendlier implements multi-level undo and redo – allowing you to undo or redo all the commands entered in one buffer (or BufferView). Note that you may have more than one BufferView open on a single physical file (or FileBuffer) at any one time.

**Navigation**

Multiple files can be opened simultaneously in Friendlier. You can open files by using ALT+O and using the browser metaphor with the cursor keys to select a file to open from the filesystem. Once you’ve made a selection you will be asked where to position the new BufferView. Again use the cursor keys to choose a position relative to the current BufferView and the file will be opened at that position and your eye position will move to view the newly opened file.

Create a new BufferView (with no associated FileBuffer) by using the ALT+N combination and selecting a direction for it. You can also copy a BufferView by using ALT+B.

You can navigate between BufferViews by using the F1 and F2 keys to cycle through them in creation order. Alternatively if you know the position of the next BufferView you can use the ALT+cursor keys combination to move to a BufferView next to that currently selected.

**BufferViews and FileBuffers**

Friendlier implements a one-to-many relationship between buffers (BufferViews) and files (FileBuffers). A FileBuffer represents a physical file on the hard drive of your computer or it could be a physical file in the cloud or on a remote server – the point being that a FileBuffer is something that is written to and read from. When Friendlier starts and the default project contains a FileBuffer it will attempt to load the physical file (or at least part of it) into the cache of the FileBuffer.

A BufferView represents a screen within the Friendlier window and has one or zero physical files (FileBuffers) associated with it. A BufferView maintains its own position in the file independent of the FileBuffer and has its own cursor position, highlighting, search term(s), background colour and other parameters. This means there are multiple ways of viewing the same or different parts of the same file.

BufferViews and FileBuffers metadata are all stored automatically whenever the user quits Friendlier. Therefore Friendlier keeps a current snapshot of the project you are working on at all times. This means that the next time you start Friendlier you’ll appear at the position and over the file you were last editing.

Note that if the project file becomes corrupted then you’ll need to rebuild your project from scratch.

**Default Project**

Friendlier uses a persistence mechanism to serialise FileBuffer and BufferView information in what is called the default project. The default project is currently the only project that Friendlier supports and can be found under the Application Data folder of your Roaming profile as a Windows user. Look for the Xyglo/Friendlier directory. In the future multiple projects and editing of the project will be supported – for the moment FileBuffers can only be added to projects. BufferViews can be removed and ‘closed’ by hitting ALT+C on them.

**Full Screen and Windowed Mode**

In theory Friendlier can displays as many views on files (BufferViews) as you would like – but in practice this is limited to how far away from these views you position your ‘eye’. You can zoom out from the current BufferView by using the F8 key. This will take your level of viewing up to where you can see four BufferViews on the screen simultaneously. It is them possible to rotate around the currently selected view using the F9 and F10 keys. You can carry on zooming out or you can use the F7 key to zoom in again.

**Keyboard and Regional Support**

At the moment there is no regional support in Friendlier and only limited keyboard support. Alternative keyboard languages will be supported at a future time.

**Saving and Quitting**

You can quit Friendlier or from within a sub-state by using the Escape key at any time. If the user tries to quit without saving a modified BufferView then a warning will be posted and an opportunity to save the file will present itself. Likewise when multiple files have been modified there is an opportunity to save all these files individually before quitting.

**Builds, Logging and Tailing**

Friendlier provides some integration with external build tools. You can specify an external build tool and build options that would for example point to a makefile and a make command. This is bound to the F6 key which will launch the build externally and capture feedback in the form of standard output and standard error files.

When a build is started Friendlier will fly to the standard output log as default so that you can keep an eye on build progress. If the build fails you will be flown to the error log to view the problems. Note that standard output and error logging utilises wrapped BufferViews which are read-only. These views are said to ‘tail’ the error and output logs and provide always the latest entries at the bottom of them.

**Configuration**

Build commands and logging directories along with other configuration information can be modified by hitting ALT+G and then selecting the entry to edit. Note that currently this editing capability is quite limited although functional.

**User Feedback**

Friendlier provides feedback in several areas on the screen. There is the area to the right of the filename at the bottom of the editor which provides information on which hot keys are depressed as well as any other information or requests. This area displays a scrolling message if the message is longer than the space available. There are also flying banners which can be seen as splash screen and at build time.

**Project information**

Information regarding number of files, length of files, time spent editing project and total project time are kept and can be displayed by hitting ALT+I.

**Keyboard commands**

|  |  |
| --- | --- |
| Keys/Combination | Action |
| F1 | Cycle down through buffer views |
| F2 | Cycle up through buffer views |
| F3 | Search again |
| F6 | Build |
| F7 | Zoom out |
| F8 | Zoom in |
| F9 | Rotate anticlockwise around group of 4 |
| F10 | Rotate clockwise around group of 4 |
| F11 | Full-screen mode |
| F12 | Windowed mode |
| ALT+N | New buffer view on new buffer |
| ALT+B | Copy existing buffer view on existing buffer |
| ALT+O | Open file |
| ALT+S | Save (as) file |
| ALT+C | Close BufferView |
| ALT+H | Help screen |
| ALT+G | Settings screen |
| ALT+Z | Undo |
| ALT+Y | Redo |
| ALT+A | Select all |
| ALT+F | Find |
| ALT+[number keys] | Jump to a numbered BufferView |

# Introduction to the Brazil Framework

To gain an understanding of the Brazil Framework it is useful to have a little history to Friendlier and the technical landscape in late 2012.

Blah blah blah..

## Brazil Background

The Brazil framework has been born of the desire to have my own 3D and interaction framework which I can port to C#, Java, Objective C and open up the understanding of these platforms in terms I can understand from a familiar viewpoint. If I fix my frame of reference to be the Brazil framework then I can lash new pieces to this framework assuming I understand their cross –platform relevance and their efficacy.

I started writing this in C#/XNA because after a little research in late 2011 it became clear that this was the easiest way to achieve my short term goals in 3D land. Therefore the basis of Brazil is XNA related – it borrows some of the constructs and classes and helper methods which make developing for XNA simple.

## Brazil Concepts and Architecture

A Brazil application is defined by specialising the class BrazilApp. An app can exist in multiple States. Actions can be created and assigned to States which point to Targets. In this manner events from keyboard or mouse (and eventually other input forms) are parameterised and stored at the BrazilApp level.

In addition to States, Actions and Targets which only have notional value to the user we also define Components which have physical (well, visible) value to the user. In the default case Components are 3D representations of actors within the BrazilApp World – indeed a member of the BrazilApp class is the BrazilWorld where the visible Components exist.

# Developing C++ Applications in Friendlier

# Developing C# in Friendlier

# Developing Brazil Applications in Friendlier

# Troubleshooting

If you have any problems getting Friendlier to install or run this section will provide you with some tips.

**Problem:**

You try to run Friendlier but you get the following message:

"Friendlier encountered a problem: Could not find a Direct3D device that supports the XNA Framework HiDef profile."

**Tip:**

This means that your graphics device is not support by Friendlier. Friendlier system requirements state that:

“You will need a graphics card that supports, at minimum, Shader Model 1.1 and DirectX 9.0c.”

For a more technical explanation of the requirements for XNA (the toolkit that is used to write Friendlier) you can see these link:

<http://blogs.msdn.com/b/shawnhar/archive/2010/03/12/reach-vs-hidef.aspx>

<http://forums.create.msdn.com/forums/t/57927.aspx>

# Contact Information

For further information on Friendlier or the Brazil Framework and for licencing opportunities please contact:

[info@xyglo.com](mailto:info@xyglo.com)

For the latest information and to download Friendlier please see:

<http://www.xyglo.com>

You can also follow us on twitter @xyglo or on Facebook:

<http:///www.facebook.com/xyglo>