“Friendlier” – Experience Development Environment
Overview and Release Notes Version 1.1
December 2012

1 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 an interface for certain graphical elements defined in Xyglo's *Brazil Framework*.

Brazil is the technology layer which underpins Friendlier. Brazil is intended to be 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 experiences be they games, presentations or something else.

If you want to skip the boring stuff about editing and actually find out what's going on here I recommend the chapter on the Brazil Framework. It's more interesting, it's still incomplete, but it's a start.

2 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. Friendlier is based around a text editor standard and borrows many features common to other established editors 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. Friendlier is 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 for you to create cool things and we believe that Friendlier in combination with the Brazil framework is it.

2.1 Future 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 wit:

- There are a lot of target platforms out there which have well documented APIs and are tasty targets for application development. Windows, Mac, Android, IOS, Linux etc.
- The APIs and the APIs that build on those APIs move quickly and can be discarded as fast as they appear.
- The underlying hardware moves a lot more slowly and so lower level APIs can stay more rigid (OpenGL, DirectX, Mouse, Keyboard, Kinect, Touch etc)
- We still need to build things.

So how do we build things?

- You can just use a proprietary low level API but that could be hard.
- You can use a third party API but you'll have to pay a licence fee, still have to invest to learn the tool and then you get locked into relying on a single vendor.
- You can use a free/open source option but you still have to learn lots of stuff.

So what do you want to do?

- Games, interactive presentations, animations.

What platforms do you want to target?

- All of them.

How much effort do you want to spend?

- Zero.

Ok, you've come to the right place. This is what Friendlier and Brazil were designed for. Zero effort doing stuff. That's where we're going. This document tells you where we are, or actually where we were a few months ago. Things are moving fast and I thought I'd better get this version of the document out before it's completely out of date.

Richard Bown
December 2012

3 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.

3.1 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

4 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. See the Introduction and the bit on "Future History" for more information. There is some more history and opinion below.

4.1 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.

4.2 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.

You can see in the following class diagram the associations between the classes. BrazilApp acts as the abstract base class for an application. BrazilApp includes a collection of States, Targets and Components – but Components are themselves abstract and have to be specialised. A BrazilApp also has a BrazilWorld which will define some limits and some things that might prove useful to have over the entire world such as gravity. You can also see the ViewSpace class which is a wrapper for the implementation layer of the graphics. At a basic level this is a very good summary of the Brazil concept.

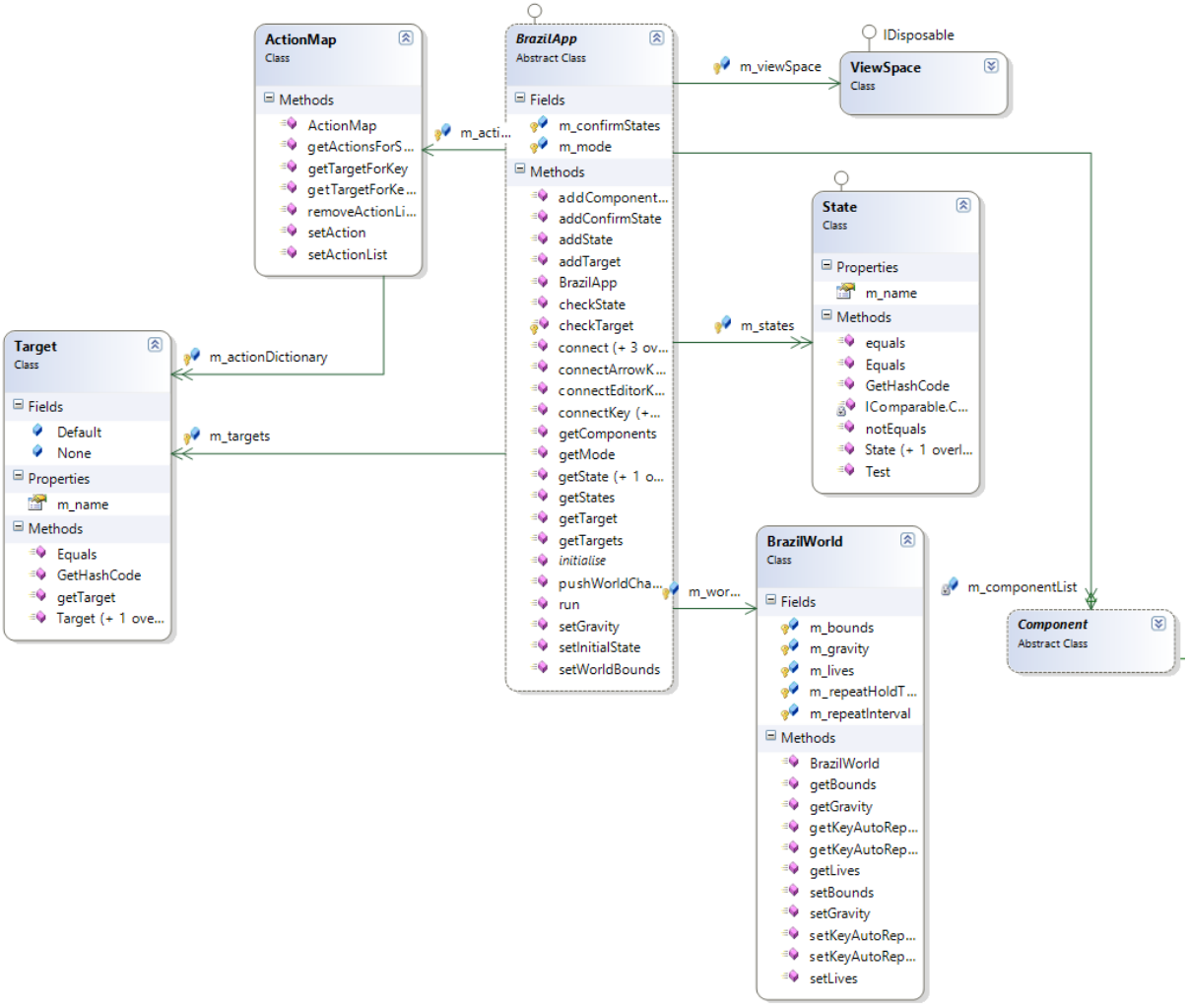


Figure 1 – Overview Class Diagram of Brazil Framework

For example, if you want to create a new application based on the Brazil framework you firstly need to specialise the BrazilApp class. Then you will use addState, addTarget and addComponent calls to populate the BrazilApp with those entities. Finally you can connect the States and Actions to Targets and Components.

Components must exist within a State. Actions combine with States and are mapped to Targets. Targets can be States in their own right and therefore can have more Components associated with them. Targets don't necessarily have to be States but of course if they aren't States then they have to be supported and interpreted somehow by Brazil – therefore there are a series of built-in States which are not explicitly defined.

Therefore

4.2.1 Built-in States

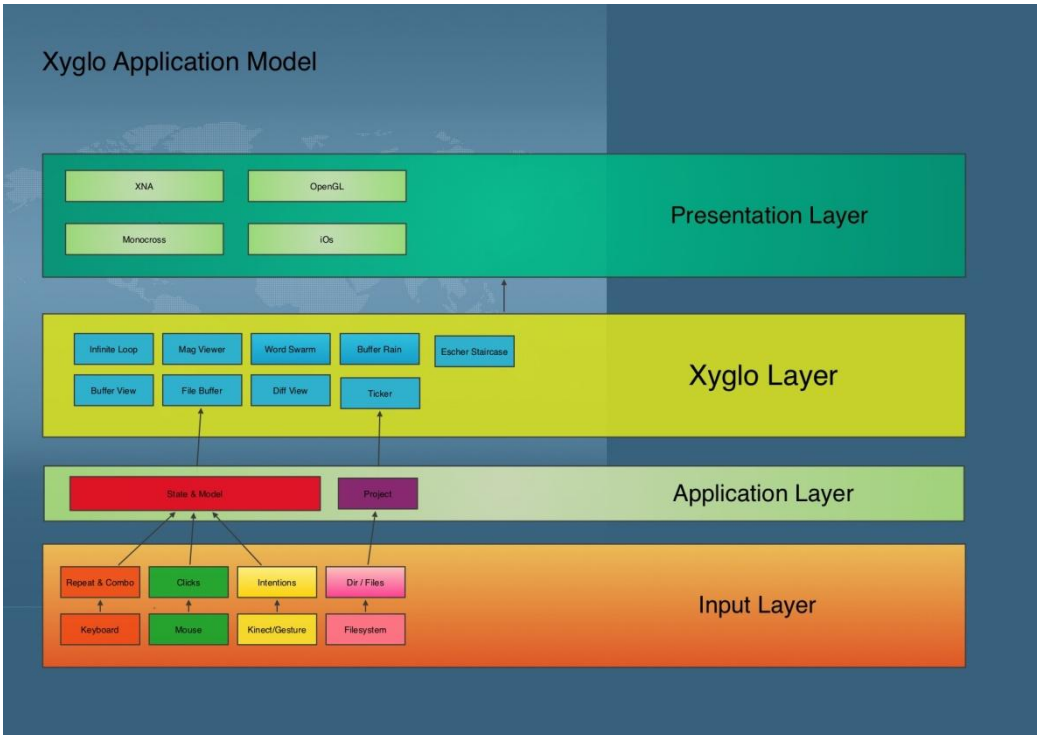
List of built-in states to be supplied.

4.2.2 Targets

4.2.3 Brazil Components

4.3 Graphics Architecture

Brazil is about gathering input and controlling 3D graphics as it's aimed at those wanting to make games or produce compelling user experiences. Therefore Brazil has to talk to the technology layer that is available on the target device and to do that we implement a variety of graphics and input interfaces which talk through the Brazil layers when it comes to implementing components and routing messages.



5 Developing C++ Applications in Friendlier

Take from xyglo.com.

5.1 Setting up Friendlier for Building C++ projects with mingw

To build C++ on Windows you'll need to download a compiler that is able to be run from the Windows command line. Friendlier is configurable so that you can point it to the build command and the Makefile that is used to build the project and it will capture the standard output and standard error from this process within a BufferView in Friendlier. Then you can use these errors/warnings to zap directly to source file lines that have issues by double clicking
on these error lines.

5.1.1 Prerequisites

To start with then you will need to download a compiler. An example of a great free command-line compatible compiler is in the MinGW (Minimal GNU for Windows) package. This package contains the g++ (GNU++) C++ compiler which is suitable for compiling most C++ you care to find. You can get download instructions for MinGW from here:

http://www.mingw.org/wiki/Getting_Started

Alternatively you could download the QtCreator package (from Nokai) which includes a mingw distribution which you can then reference. In the case of building Qt apps (and in this case I have been building Rosegarden for Windows) you may find it useful to install QtCreator as it comes with all the Qt libraries of a compatible level too. I would also highly recommend QtCreator (or the Qt libraries at least) because they come with qmake which is a nice simple way to create and maintain C++ projects. This guide will assume you've got MinGW and Qt/QMake installed.

Additionaly QtCreator is a very fine (and free) IDE for developing C++. You can certainly use it in addition to Friendlier!

Once you have MinGW installed - either standalone or through QtCreator then you need to ensure that your PATH environment is set up to find the build tools.

Open a command window:

a) Click on Start Menu
b) Type cmd and hit return
c) Type 'set' and look for the PATH variable ensuring that it includes a path to the ming\bin directory.
d) If not set then please add this through Control Panel\System and Security\System and then Advanced System Settings
and then 'Advanced' tab and the 'Environment Variables'

You can test that the mingw32-make.exe is in your path for example by opening a cmd window and typing this command. The command should be found and should complain that no Makefile has been found.

If you want to build an existing MinGW compatible project then you can download and install the source code at this point.

If you want to start a new C++ project from scratch then you should create a new empty directory to hold you source code. For the following example we will assume you've got an empty directory for your project at C:\MyCppProject

5.1.2 Configuring Friendlier and Building

You can now start Friendlier. Note that any configuration items you modify will be stored automatically when you shut down Friendlier and restart it.

When Friendlier has started:

1. Ctrl + G to open configuration items

2. Select BUILDDIRECTORY and set to the build directory to that which you require for the solution.

i.e. C:\MyCppProject

Tip: You can cut and paste values into the configuration value which can save some typing

3. Firstly we'll need to create a project file for your new project. If Friendlier isn't looking at an empty BufferView then you can press ALT+N together to open a new one.

4. You can set up a new project file by following the examples here for qmake:

http://doc.qt.nokia.com/4.7-snapshot/qmake-project-files.html

and:

http://doc.qt.nokia.com/4.7-snapshot/qmake-common-projects.html

For our example you can enter the following information in your empty BufferView:

```
TEMPLATE = app
DESTDIR  = C:\NewCppProject
SOURCES += HelloWorld.cpp
CONFIG  += console qt warn_on release
```

5. Now save this BufferView as a file. Hit ALT+S (for Save) and you'll be prompted for a location. Use the cursor keys to navigate your file system to the C:\MyCppProject directory. Once there then type the name of the project file "HelloWorld.pro" and hit return. You'll see the file name gets added to the directory location at the top of the screen. When you hit return the file will be saved.

6. Now we need to create our HelloWord.cpp file. Open a new BufferView by hitting ALT+N. You'll be prompted for a position to place this new BufferView - use the cursor keys to selection a location.

7. Now you can create the contents for this file:

```
#include <iostream>

int main(int argc, char **argv)
{
  std::cout << "Hello World" << std::endl;
  return 0;
}
```

and save this file as C:\MyCppProject\HelloWorld.cpp

8. Now we can configure and run qmake for this project. Open the config screen with ALT+G.

9. Select ALTERNATEBUILDCOMMAND and hit return. Give this the path to the qmake command on your system. For example on my system Qt is installed under C:\Q\ and the path to qmake is:

C:\Q\Desktop\Qt\4.7.4\mingw\bin\qmake.exe

You'll also need to add the path to your project file and what to do with it. We want to generate a Makefile from this project file so we set ALTERNATEBUILDCOMMAND to read like this:

C:\Q\Desktop\Qt\4.7.4\mingw\bin\qmake.exe -makefile C:\NewCppProject\HelloWorld.pro

10. Go back to editing mode by hitting Escape. If you've set this ALTERNATEBUILDCOMMAND up correctly you should then be able to hit F7 in Friendlier and qmake will run. It should create three Makefiles (Makefile, Makefile.Debug, Makefile.Release) plus a debug and release directory under C:\NewCppProject

11. Now configure your BUILDCOMMAND. Hit ALT+G and set it to something like:

C:\Q\mingw\bin\mingw32-make.exe -f C:\NewCppProject\Makefile

12. Get back to editing mode (Escape to come out of Configuration mode) and then hit F6 and you should see your HelloWorld.cpp building. Friendlier will switch to the standard out log so you can see the build progress.

13. If the build completes successfully you can now run your program. Open a cmd window and change to the C:\NewCppProject directory and run your HelloWorld.exe application.

14. If your build doesn't complete successfully you will be shown the standard error output window (in red) which will give you clues as to why the build failed. If you need to modify your project (.pro) file again you will also need to hit F7 again to re-run qmake. This will regenerate the Makefile afresh and then you can hit F6 to rebuild the source code.

In this manner you can quickly create a C++ project from scratch. Please study the qmake documentation to see how you can add libraries and other include files to your C++ project along with other resources. With Qt libraries you can also build native GUI applications in Windows C++. Have fun!

6 Developing C# in Friendlier

There will be a way to do this at some point.

7 Developing Brazil Applications in Friendlier

There will also be a way to do this – quite soon.

8 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

9 Contact Information

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

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