*F*lexible *I*nterface *R*endering *E*ngine

for J2ME

by padeler

padeler@bluebird.gr

Documentation Draft

v0.1

1. Fire

Fire is a GUI Library that aims to provide a set of easy-to-use extendable components to j2me developers. The basic set of Fire components offer all the functionality of the j2me GUI components (Forms, Items etc)plus a much more appealing user interface, themes, animations and better component lay-out. Also Fire does not depend on device-vendor specific parameters to layout its components on the screen, thus creating interfaces that look the same on different phones and screen sizes.
A design requirement of Fire was to make the transition from traditional j2me GUI components as easy as possible for the developer. With that in mind, the concepts of the Display singleton and the CommandListener exist in the Fire engine too.

1. Engine Requirements

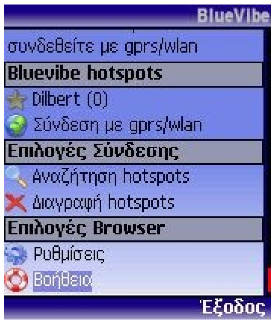
   a. CLDC 1.0
   b. MIDP2

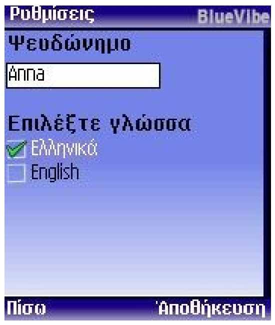
2. Engine Features

   a. Fire UI Components provide all the functionality of MIDP UI Components
   b. Themable UI (themes in the form of a .png image file)
   c. Animated Components
   d. Keyboard and touch screen driven interface
   e. Popup menus support
   f. Stackable popup menus/windows
   g. Full Screen Support
   h. Ticker like text areas
   i. Listener based events
   j. Easy transition from Form/Items (midp2.0) based interface.
   k. Utility class FireIO, provides easy to use I/O methods
   l. Utility class Lang, provides easy to use i18n support.

Fire stands for Flexible Interface Rendering Engine. It was originally developed as part of the Bluevibe platform (www.bluevibe.gr), and it is mainly used for the various flavors of the Bluevibe Browser. Its current development status is stable. In the feature versions there are plans for more UI Components (as an external contributions package) and a better theme management system.

2. Core Engine Components




Ρυθμίσεις BlueVibe

Ψευδώνυμο

Anna

Επιλέξτε γλώσσα

[✓ Ελληνικά]
English

Πίσω Αποθήκευση

2.1 FireScreen
FireScreen is a singleton just like the Display. It can draw Panels and Popup-panels. It draws the theme elements and handles the animations of transition from one panel/popup-panel to another.

Only one Panel can be shown on the FireScreen at any time, but any number of popup-panels can be stacked above it. Only the top-most component (panel or popup) is active, and receives events. For simplicity we will call both popups and panels as panels, because (apart from the decorations/animations) they are handled the same way by the FireScreen instance.
The FireScreen is responsible for sending periodically clock events to its top level panel (the active panel) in order to update animations. User input (key and tap events) is also send to the active panel.
The FireScreen loads its theme parameters from a given 50x17 png image. This image is divided into regions as shown bellow:

Location	**Description**
Area (0,0)-(49,14)	Logo of top right corner
Line 16 (0,15)-(49,15)	Background gradient
Pixel (16,0)	Background color
Pixel (16,1)	Label color
Pixel (16,2)	Scrollbar color
Pixel (16,3)	Link color
Pixel (16,4)	Selected link foreground color
Pixel (16,5)	Selected link background color
Pixel (16,6)	Rouler primary color
Pixel (16,7)	Rouler secondary color
Pixel (16,8)	Scroll Rouler primary color
Pixel (16,9)	Scroll Rouler secondary color
Pixel (16,10)	Scroll Rouler third color
Pixel (16,11)	show background image(if(val==0xFF000000) showBgImage = false)
Pixel (16,12)	show gradient. (if(val==0xFF000000) showGrad = false)
Pixel (16,13)	Primary filled row color
Pixel (16,14)	Secondary filled row color
Pixel (16,15)	Border color
Pixels (16,16) - (16,33)	RFU
Pixels (16,34) - (16, 49)	Top and botton decorations gradient

For a description of the public methods of FireScreen please refer to the javadoc and to the sample application.

2.2 Component
Minimal drawable primitive. All elements of the Fire GUI are Components. Every component can have Commands assigned to it. An action is fired if the component receives a “fire” event (key press or tap). A Component is traversable if it has a Command assigned to it. When traversable a component receives user input. A Component is responsible for changing its visual status when it is traversed. If a Component is animated (its isAnimated() method returns true) it will receive clock events periodically in order to update its animation. A component can contain other components. A component action can cause a panel to be displayed/hidden. The component should send this event to the FireScreen using the FireScreen.getScreen().showPopup() and closePopup() methods.

2.3 Panel
A container for Components, a Panel is also a Component. The panel is responsible to pass messages to and from the components. The panel can draw any number of Components of fixed width (see Row). Every new Component that is added will be placed on the bottom of the last one. The panel is also responsible for handling mobile softkey events. Commands and popup panels can be assigned to the left and/or the right softkey and the panel will display the information accordingly. It implements the scrolling when the total height of the contained components is bigger than the screen height, and draws the theme decorations. The default decoration are a top and bottom bar and a scrollbar area. A common decoration image is created by the FireScreen on theme initialization and can be retrieved by a panel, but this may change in feature versions.

2.4 Row
A Row is a multipurpose fixed width component that can display elements horizontally. It can show an image , a string or both. It can be associated with one Command. In combination with the Panel which arranges Components vertically, the GUI designer can create a variety of interfaces. A row can also act as a TextField for user input or as an (in)visible separator between other Components on the panel.

2.5 PopupPanel
A Popup panel is a lightweight version of the Panel, it doesn't draw theme decorations and it can be smaller than the screen width/height. The popup will pack itself according to its Components. It can have scrollbar if the total height of the contained components is larger than the assigned max height. Because it extends Panel, it can work with any type of (fixed width) component. The Panel's and the PopupPanel's functionality are probably going to be merged in feature versions, in order to provide the same functionality from only one class.

2.6 FTicker
A Ticker is also available in the Fire Engine. It can be added to Panels and works the same way as the j2me ticker.

2.7 FString
Text primitive. Given a Font and a max width the Fstring instance wraps a string in order to be displayed correctly in a Component. It replaces all non printable and white characters with ' ' (space).

2.8 FGauge
A simple animated “busy” indicator. It is displayed on the bottom of the screen (on the middle of the bottom bar) when the setBusyMode(true) is called on the FireScreen.

2.9 FireIO

Provides an abstraction for I/O. It provides methods for reading and writing to records, and making http requests. It can download Image files from a given http location transparently to the system when requested, and store them in the recordstore for later reuse. It also caches images loaded using the getLocalImage() method.

2.10 Lang

Lang is a utility class for internationalization. It has a default language “EN”. It assumes that the keys are in English, and returns the translation according to the bundle that is loaded. If a string does not have a translation in the language bundle it returns the given key. For more details please look in the javadoc.