



JM-Mobile

Java Multimedia for Mobile

User's Guide

JM-Mobile Editor version 1.2x

Copyright ©2008 JM-Mobile.com all rights reserved.
JM-Mobile.com has intellectual property rights relating to technology embodied in the product that is described in this document.

Contents

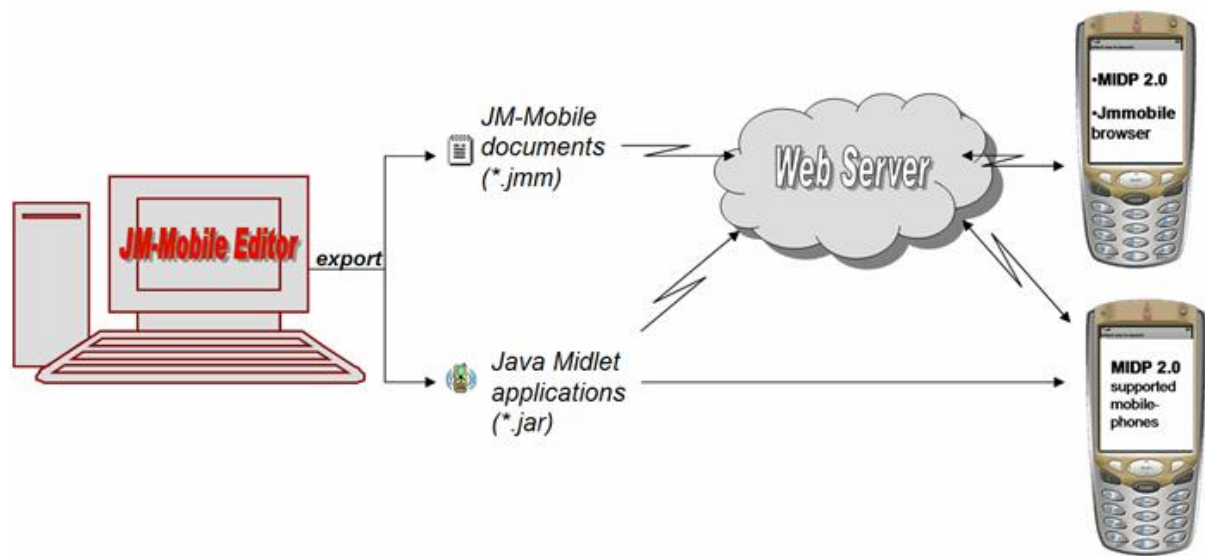
Introduction	4
Overview.....	6
Interface	8
Menu bar	8
Tool bar	11
Structure View	12
Spatial Layout View.....	14
Spatial Attribute View	16
Timeline View.....	18
Temporal Attribute View	19
Runtime active condition.....	20
Exports	22
Export to Midlet Application.....	22
Export to *.jmm Document	23
Setting	25
Mobile Font Setting	25
Project Structure	26
XModel.....	27
VARs	30
AdditionalRes	32
Media	33
Text	33
BitmapText.....	35
Image	35
Geometry	36
Audio	37
Video	38
Event and Behavior	39
KeyAccess.....	39
Composite	43
Temporal Group.....	43
Spatial Layout Group.....	44
Image List	45
Text List.....	45
Slideshow	46
Menu	46
Form.....	48
XMenuTemplate	50
DMenuutemplate.....	53
Scene.....	56
Performance Hint.....	57
Contacting Us.....	58

Introduction

Welcome to Java Multimedia for Mobile - JM-Mobile Editor Version 1.2.5

JM-Mobile Editor is a graphical multimedia authoring tool. It enables non-programmers such as designers to produce and develop complex and sophisticated multimedia presentations for the mobile-phones that support Java MIDP2.0. Author has a built-in graphic engine, allowing to easily do GUI, time and navigation scenarios edit and then to export to final application/document for mobile-phones.

The figure below presents the production scheme of the JM-Mobile Editor:



The multimedia presentations edited by the JM-Mobile Editor can be deployed in two ways:

- the *java Midlet applications (*.jar)*, that can be downloaded directly into the mobile-phones or can be uploaded on the Web servers to distribute largely ;
- the *JM-Mobile Documents* (a private format), that can be uploaded on the Web server, then can be downloaded and playback by the mobile-phones that are installed our *JM-Mobile Browser*. It can be also called from another JM-Mobile Application or another JM-Mobile Document by the hyperlinks. The JM-Mobile Document can be also added into our Bluetooth Server to distribute through Bluetooth communication.


The *JM-Mobile Document* deployment is more flexible than the *java Midlet application* deployment. It allows you and your clients updating easily new contents of the application. The *java Midlet application* deployment requires that your clients must reinstall the application each time its content is updated.

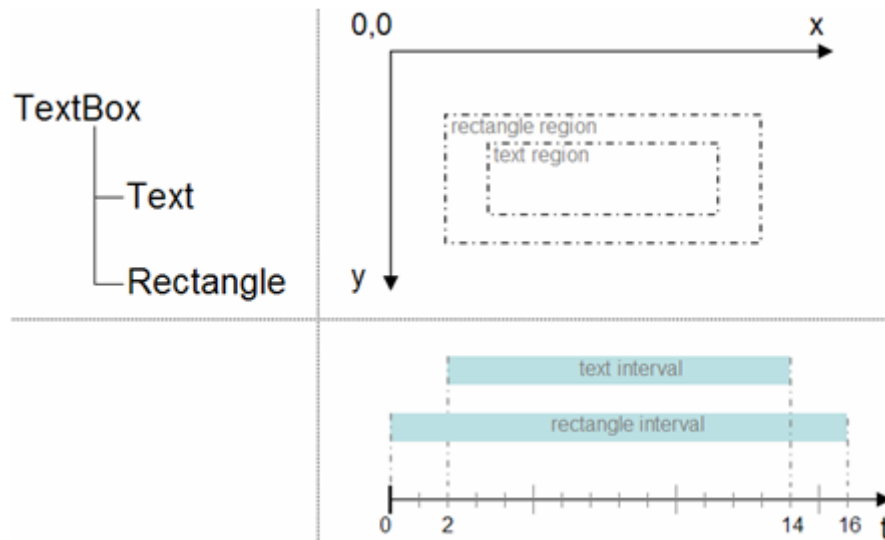
The *JM-Mobile Browser* can be downloaded from our site web <http://www.jm-mobile.com>.

The *JM-Mobile* supports the command line mode, type *jmmobile.exe -h* in the command line window for more detail. The command line mode is important to create an automatic JMM application/document exportation system.

Overview

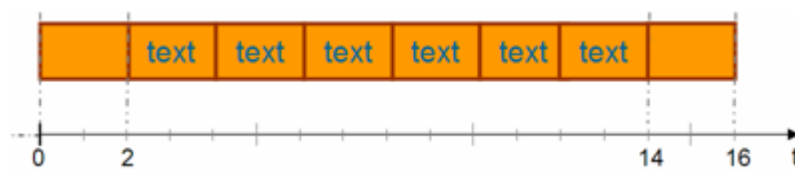
The JM-Mobile Editor relies on a multimedia presentation model that contains three principal dimensions: *Structure*, *Spatial* and *Temporal*.

Let's take a simple example of composition of a  through these dimensions, it should look like as following:



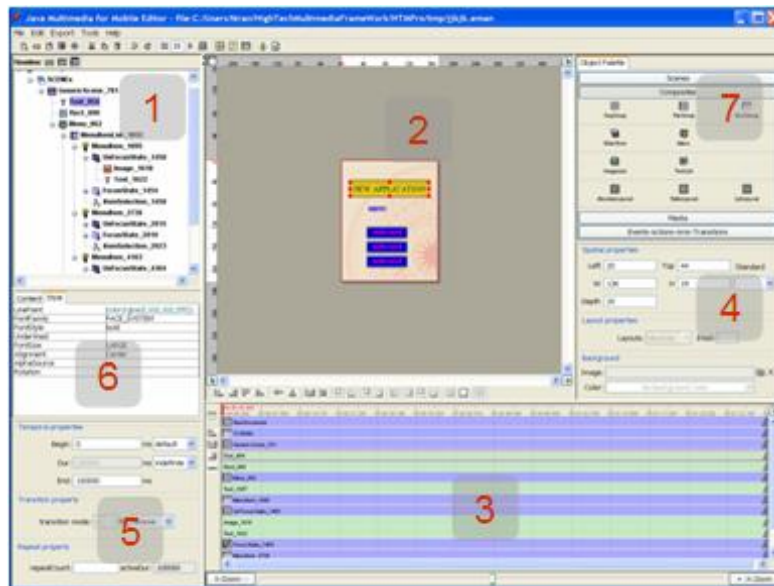
The TextBox is composed of a *Text* and a *Rectangle*, the *Text* and the *Rectangle* have a *region* and an *interval* to specify WHERE and WHEN they are displayed on the screen.

It should be noted that the integration of temporal dimension (*interval*) allows to evolve the multimedia presentations in the time, see the TextBox's presentation figure below.



The presentation time of the TextBox is from 0 to 16s. During the first 2s only the *Rectangle* of the TextBox is displayed, the *Text* is displayed at the 2nd second and is during until the 14th second then the *Text* is disappeared. The *Rectangle* is also disappeared after the 16th second.

To help to edit easily such multimedia presentations our JM-Mobile Editor provides a multi-view interface as below:



- (1) is the *structure* view, it's the most important view that can give a view global through all structure of the application. Thanks to this view the authors can easily edit, manage, and modify their applications.
- (2) is the *spatial* view, it displays the application and allows also to directly manipulate spatial location of objects.
- (3) is the *timeline* view, it shows the time scenario of application and allows also to directly manipulate the presentation time of objects.
- (4) is the spatial attribute view, it shows and allows to edit the spatial attributes of a selected object.
- (5) is the temporal attribute view, it shows and allows to edit the temporal attributes of a selected object.
- (6) is a tablepane that contents a group of the content|style|interactive attribute views.
- (7) is the object palette, it presents the objects that author can drag and drop into the spatial view.

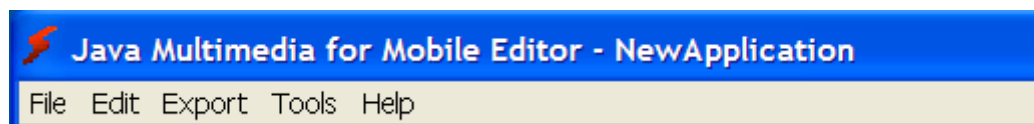
Interface

JM-Mobile Editor provides a WYSIWYG multimedia authoring interface that allows to switch between editing and playing modes to see immediately editing results. Authors then don't have to often export their multimedia projects each time they want to see or test theirs editing results.

The main interface includes:

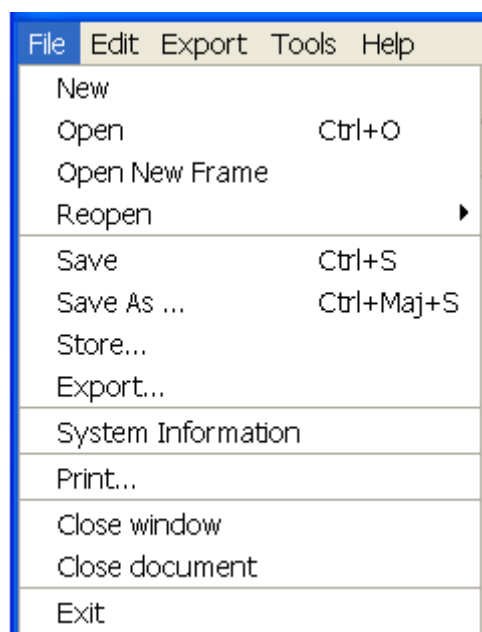
- a *menu bar*,
- a *tool bar*,
- a logic *structure view*,
- a *spatial layout*,
- a *temporal organization* (timeline) of multimedia project,
- the *attributes views* and
- a pallet object.

Menu bar



The title of application consists of the logo, the name of tool and then the name of editing multimedia project.

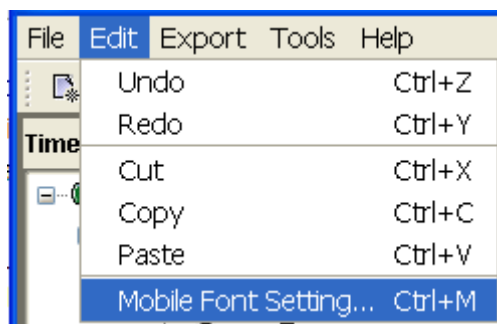
File



The File menu consists of:

- the **New** item that allows to create a new project;
- the **Open** item that allows to open an exist project;
- the **Open New Frame** item that allows to open an exist project in a new frame;
- the **Reopen** item that provides a quickly opening one of the last 10 projects;
- the **Save** item that allows to save the editing project;
- the **Save As** item that allows to save the current project into another project;
- the **Store** item that allows to save the current project and its resources (images, audio, video, etc.) into a same place, so your project will be portable.
- the **Export** item that allows to export the current project;
- the **System Information** item that allows to see the memory state of the system;
- the **Print** item that isn't implemented yet;
- the **Close window** item that allows to close a opening window;
- the **Close document** item that allows to close a opening project;
- the **Exit** item that allows to exit the application.

Edit



the Edit menu consists of:

- the **Undo** item that allows to undo until 20 last modifications;
- the **Redo** item that allows to redo a modification;
- the **Cut** item that allows to delete an object;
- the **Copy** item that allows to copy a object;
- the **Paste** item that allows to paste the copied object in to a place;
- the **Mobile Font Setting** item that allows to change/configure the mobile fonts for editing;

Export

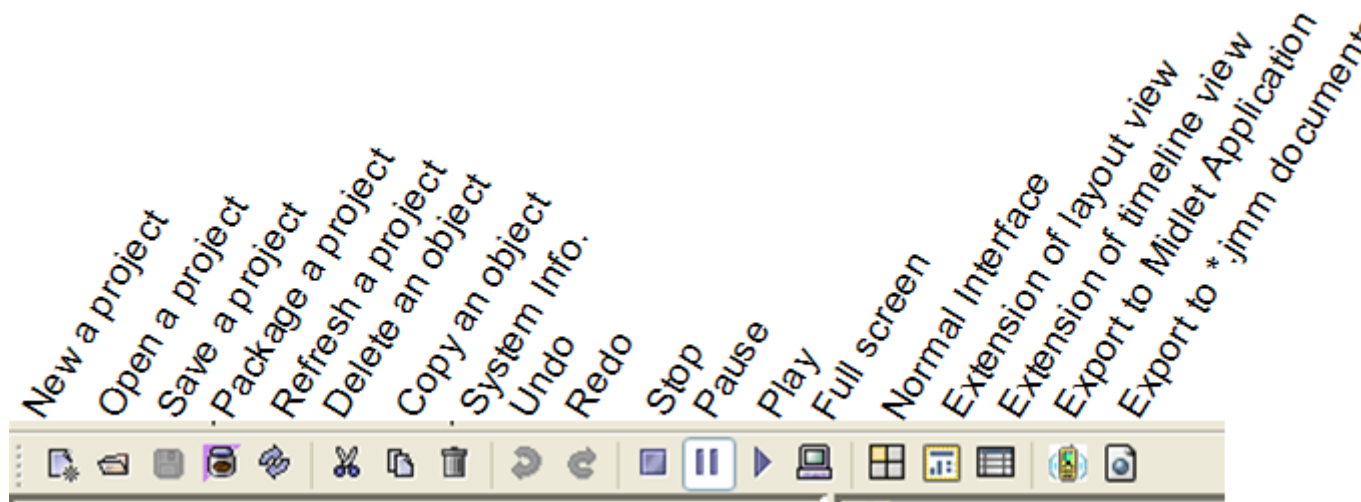
the Export menu consists of the **Show playback request** item that allows to show a dialog requesting playback the exported application.

Tools

The Tools menu contains:

- **Index Generating...** item allows to generate an *.ijm file (indexing jm-mobile documents file) from a rss file that describes a liste of jmm documents.
- **Install Key License** item that allows to choose to install a key license provided.
- **Bitmap Font Generator...** item starts a bitmap font generator tool, that allows to create the bitmap fonts and to integrate them into the editor for using.

Tool bar



Control Group

Play

This button allows to switch to playback mode to test/see the editing results

Pause

This button allows to pause to edit at any moment of the project.

Stop

This button allows to stop and unmap all objects from screen, it then sets the project ready to play from beginning.

Export Group

Export to Midlet Application

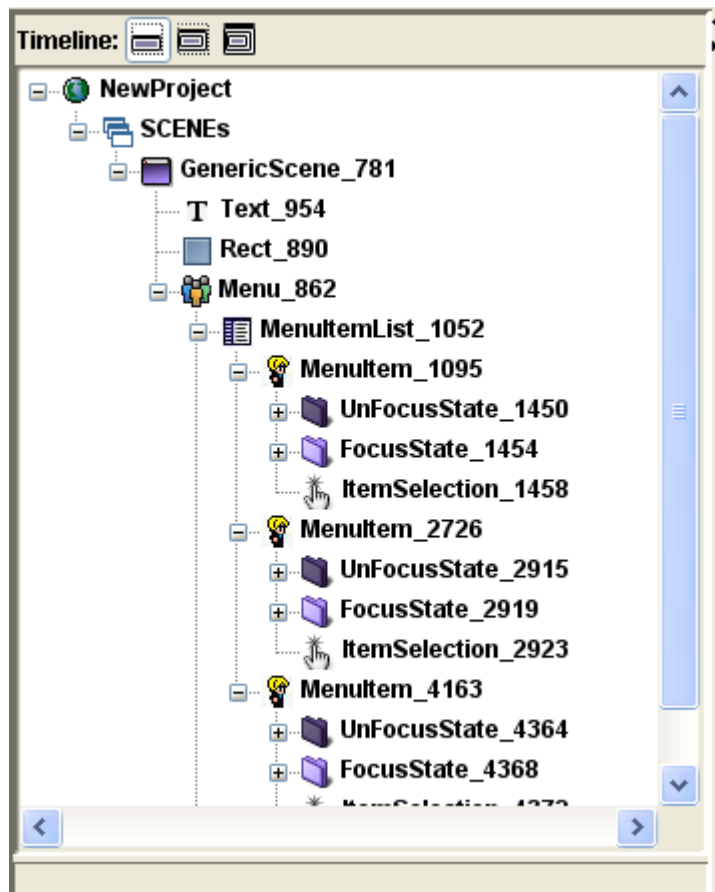
This button starts the process exporting the editing project into a Java Midlet Application.

Export to *.jmm document

This button starts the process exporting the editing project into a *.jmm document.

Structure View




The structure view shows the structure of the editing multimedia project. It allows therefore user to easily manage the editing project. It's structure view is the most important view of the system. Thanks to this view author can navigate and directly access to any element of the project structure to see, to edit or to modify it.



Timeline synchronized button group

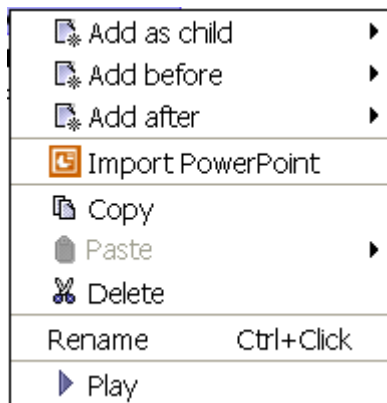


This group of three buttons allows to synchronize each selection on this structure view with the presentation of the timeline view. Three buttons accord with three synchronization levels:

- if the button  is selected, the timeline view shows only the timeline element according to the selected object in the structure view.
- if the button  is selected, the timeline view shows the timeline scenario of the Scene containing the selected object in the structure view.
- if the button  is selected, the timeline view shows the timeline scenario of whole project.

Context menu

Author can move the mouse pointer over an element of the structure view and click on the mouse right button to access to a context menu according to this element of the structure as following:



Addition group items

This menu item group can guide author to create a new element and then insert it *as child*, *after* or *before* of the context element in a coherence manner according to the *project structure*.

Import PowerPoint presentation

This menu item allows to convert a powerpoint presentation to a list of static images, then integrates this list of images into the project.

Modification group items

This edition item group allows to copy, paste or delete an element of the structure.

Rename item

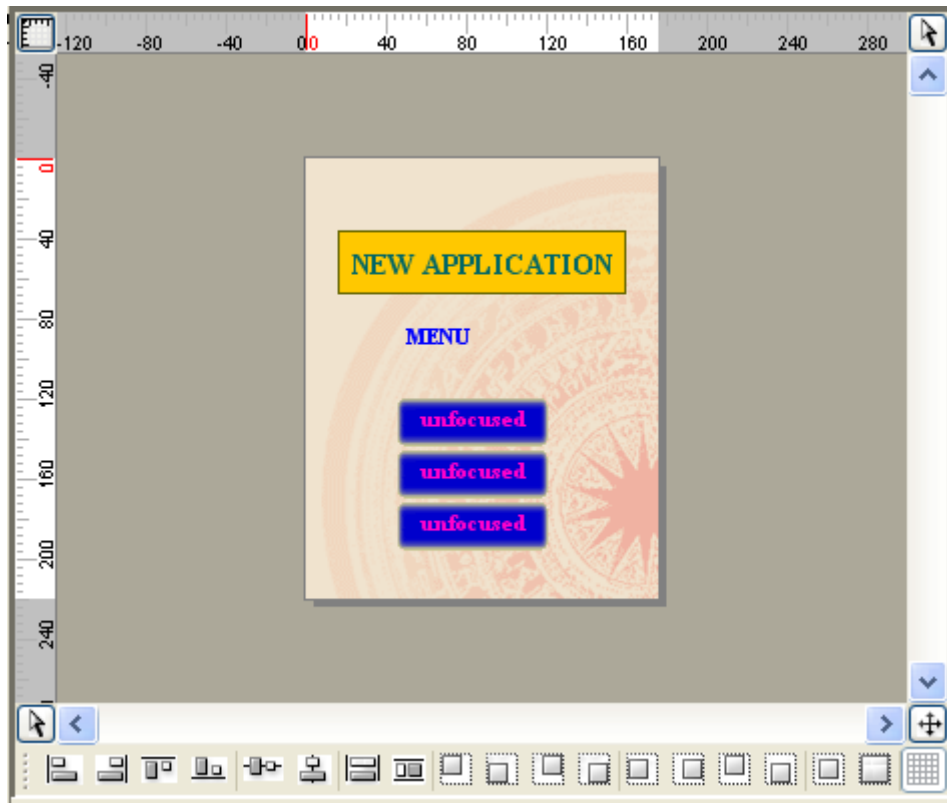
This menu item allows to rename the element of context menu.

Play item

This item allows to start playing the project at the beginning of the element of the context menu.

Spatial Layout View

The spatial layout view allows author to organize the spatial layout of project in a 2.5D space (x, y and z-index). It is also the presentation screen of the project when author change to the playing mode.



Object Region and Container

The JM-Mobile uses a spatial hierarchical layout, where the **container** can contain the others **containers** and the **object regions**.

An **object region** is a rectangle where a visible object such as a text, an image, a rectangle or a video is displayed.

A **container** allows grouping a set of containers and/or object regions. In addition, the flow container allows aligning its components in the vertical or in the horizontal direction.

When you insert a visible object, an object region is created automatically for that visible object.

When you insert a composite element such as a scene, a menu, a button, etc. a container is automatically created to group all components of this composite element. Some composite elements such as Listlayout, FlowLayoutMenu, etc. will create the flow containers.

Attention: the hierarchical layout system will help to organize easily the spatial components, but sometime it creates the difficulties in editing. Please pay attention when you edit the deep hierarchal structures as like as the MenuItem, the Button, etc.

Selection/Unselection


You can select an object region or a container by click on it. When a object region or a container is selected, the visible object or the composite element corresponding with will be also selected on the structure view.

You can select many object regions and/or containers by using Shift+ mouse click or mouse pressing dragging and releasing. When multiple object region and/or containers are selected, there is always one selection is red, this is the referenced selection, the alignment and the resize of the selected elements will use this referenced selection as the reference. You can change a referenced selection by using the Shift + mouse click.

Attention! You must be sure that the selected object region and/or container must be direct children of a same container.









You can unselect a container by double click on this container or select another container/object region. To unselect an object region you can select another container/object region, or click mouse right button then choose the “unselect all” item on the context popup menu.

Rules

There are a horizontal and a vertical rules that allows author to manipulate the regions more easily. Authors can change the scales of these rule by using the button  on the left-top corner of the view.




Spatial layout toolbar






Multi-region alignment :

- the button  allows to align all selected region by the **left** side.
- the button  allows to align all selected region by the **right** side.
- the button  allows to align all selected region by the **top** side.
- the button  allows to align all selected region by the **bottom** side.
- the button  allows to align all selected region by the **horizontal** axis.
- the button  allows to align all selected region by the **vertical** axis.
- the button  allows to make **equal the width size** of all selected region.
- the button  allows to make **equal the height size** of all selected region.

One-region alignment

This item group allows author to align the selected region with its container :

- the button  allows to align the selected region with the **top-left** corner of its container.
- the button  allows to align the selected region with the **bottom-left** corner of its container.
- the button  allows to align the selected region with the **top-right** corner of its container.

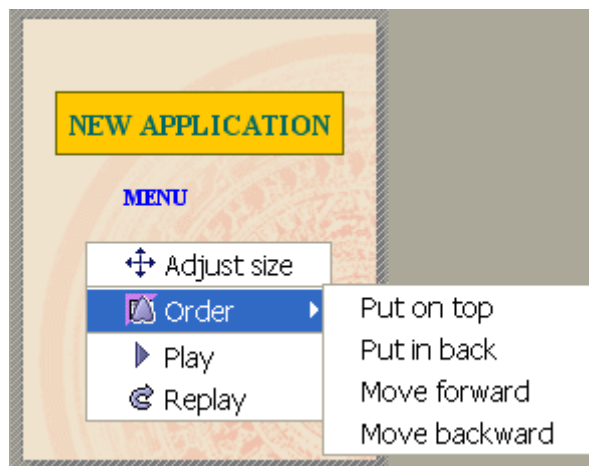
- the button  allows to align the selected region with the **bottom-right** corner of its container.
- the button  allows to align the selected region with the **center-right** corner of its container.
- the button  allows to align the selected region with the **center-left** corner of its container.
- the button  allows to align the selected region with the **top-center** corner of its container.
- the button  allows to align the selected region with the **bottom-center** corner of its container.

Grid item

This menu item allows to set visible or hide the grid on the spatial layout.

Context menu

Author can move the mouse pointer over a region and click on the mouse right button to access to a context menu according to this region as following:



Adjust size item

This menu item can allows author to fix the size of region according to the intrinsic dimension of its media content such as the original size of image.

When the selected component is a container, the size and the location of the container will be recalculated to be the smallest rectangle bounding all its children components.

Order

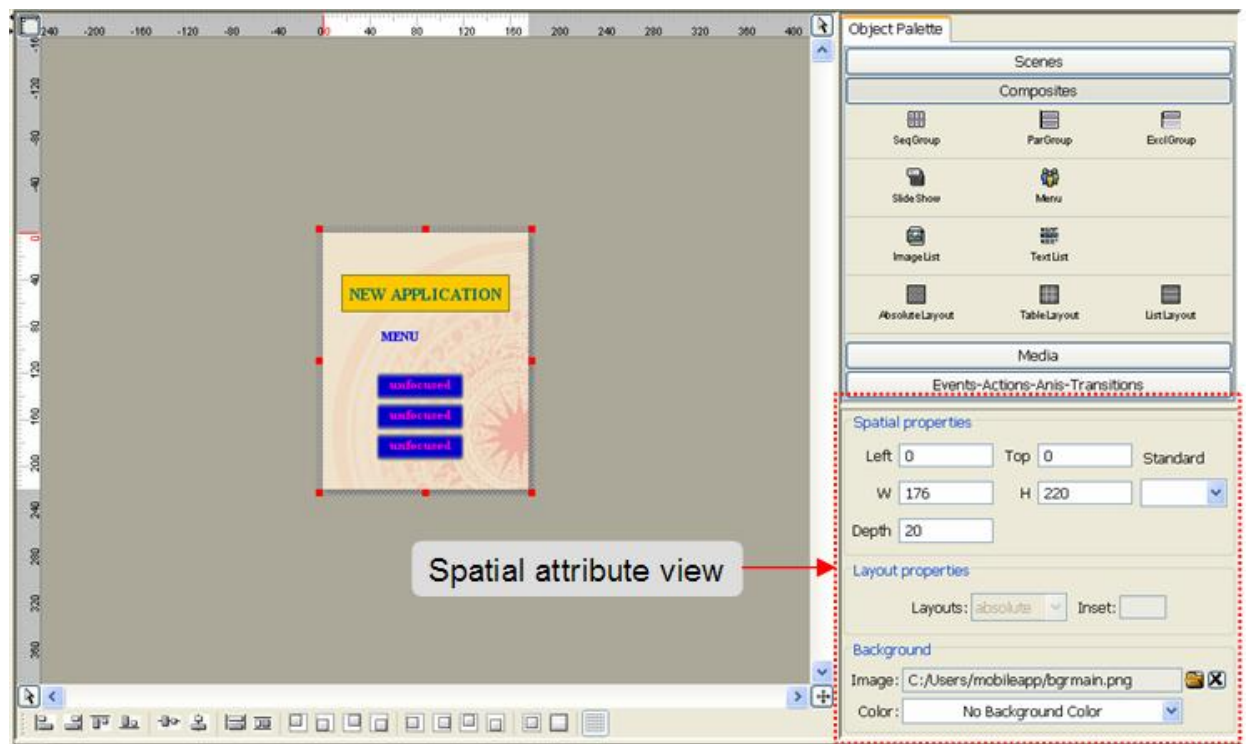
This menu item allows to change the z-index of the region of the context.

Play item

This item allows to start playing the project at the beginning of the element of the context menu.

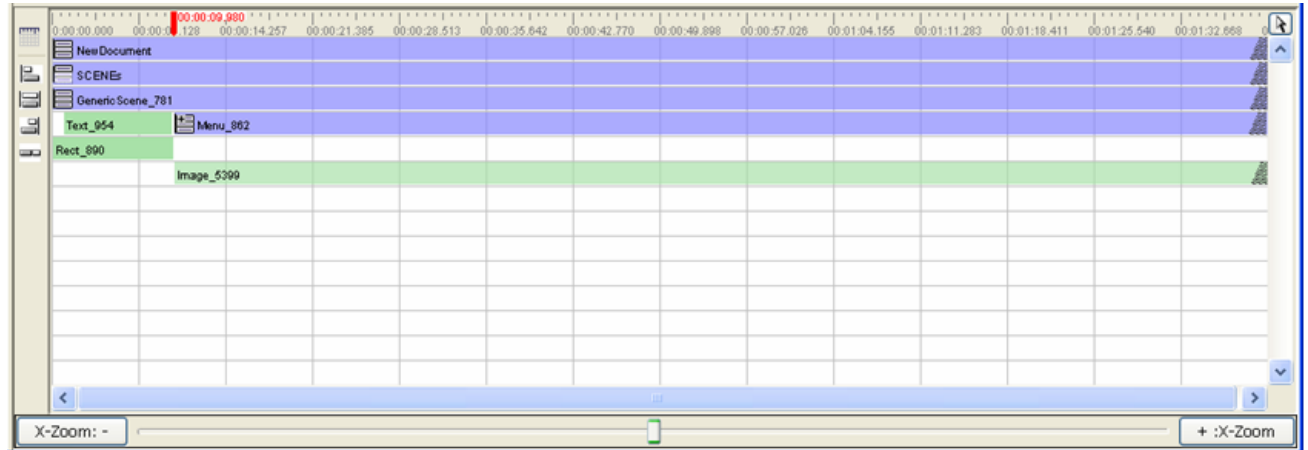
Spatial Attribute View

When a region is selected, its spatial attributes are showed on the attribute view. That gives author not only a global view on spatial information of the selected object, but also the capacity of direct modification on these spatial attributes.



Timeline View

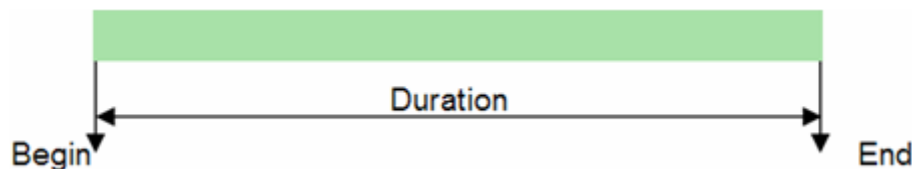
The Timeline view provides a global graphical view of the time scenario of the project or a portion of the project. Author can also directly modify the time scenario of the project through this view.



Rectangle

Each rectangle in the view presents the time life of an object in the project. It is also called *Timeline element*:




- the **left side** of the rectangle presents the **begin** time of the object;
- the **width** of the rectangle presents the **duration** of the object;
- the **right side** presents the **end** time of the object.



To select a timeline element, click on the rectangle; to modify the time attributes, drag the mouse on the rectangle or on the left side or the right side of the rectangle.

Attention! Author cannot make a manipulating-direct modification if the time attribute according to the modification is *indefinite*.

Indefinite time

-  if this symbol presents on the left side of the rectangle, that means the **Begin** time of the object is *indefinite*.
-  if this symbol presents on the right side of the rectangle, that means the **End** time of the object is *indefinite*.
- if the background color of the rectangle is **semi-transparent** , that means the **Duration** of the object is *indefinite*.

There is the dependence between the *Duration* attribute and the *End* attribute. If the *Duration* attribute is *indefinite*, the *End* attribute will also become *indefinite* and vice versa .

The *Begin indefinite* means that the object won't be automatically activated in the time; it can be activated by a couple of event and action (*StartElemAction*) from another object.






The *End/Duration indefinite* means that the activated object won't be automatically ended in the time; it can be ended by a couple of event and action (*EndElemAction*) from another object, or its parent is ended.

The *indefinite* value is very important to make the presentation of the application being interactive, that means the presentation of the application could be adapted in according to the interaction of the user.

To change a time attribute to indefinite or definite, author can click on the right mouse button then choose *Begin* or *Duration* item. Author can use the *Temporal Attribute view* to do this modification.






Hierarchical timeline

The timeline view allows to represent hierarchically the time scenario of project:

- the rectangle in the color  presents a basic timeline element. The basic timeline element always represents a media object such as an image, a text, a geometry, etc.
- the rectangle in the color  presents a composite timeline element. The composite timeline element always represents a temporal container/operator. There are three type of temporal container: parallel (*par*), sequence (*seq*) et exclusive (*excl*). Each type of container is distinguished with other by a icon in the left side of the rectangle:
 -  this icon represents a *parallel* container (*par*).
 -  this icon represents a *sequence* container (*seq*).
 -  this icon represents an *exclusive* container (*excl*).

Toolbar of the timeline view

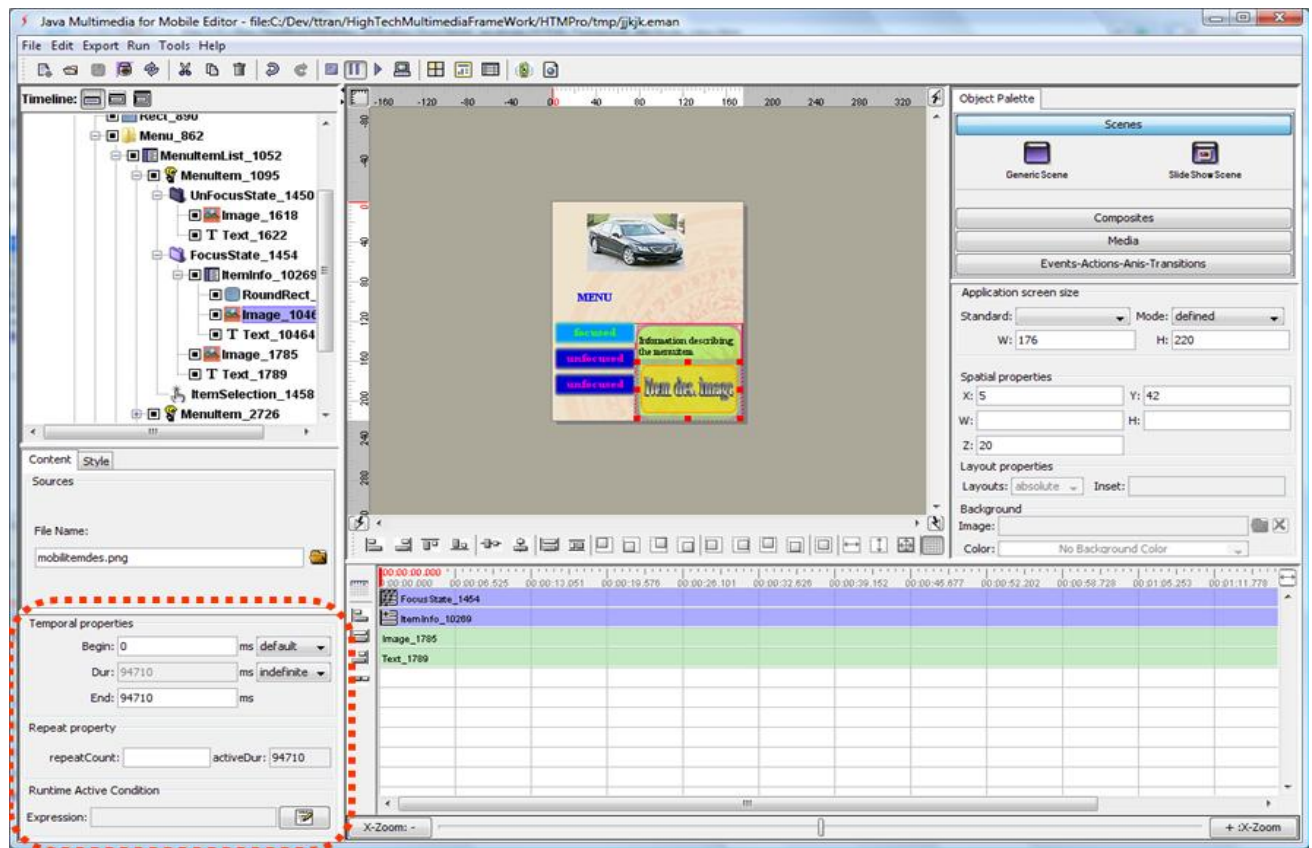
On the left side of the view is a toolbar that allows to

- change scale of the temporal rule ,
- align the rectangles representing timeline elements:
 -  this button sets equal the *begin* time of all selected timeline elements;
 -  this button sets equal the time life (i.e., *begin*, *duration* and *end*) of all selected timeline elements;
 -  this button sets equal the *end* time of all selected timeline elements;
 -  this button sets equal the *end* time of the first selected timeline element with the begin time of the rest selected timeline elements.

Temporal Attribute View

When an object is selected, its *timeline element* is selected in the *timeline view*, in addition its temporal attributes are showed on the *temporal attribute view*. That gives author

not only a global view on temporal information of the selected object, but also the capacity of direct modification on these temporal attributes.



Runtime active condition

Each element can be specified a logic **expression** attribute. The logic **expression** will be evaluated at the runtime when the object is activated. If the **expression** is **false** the object will be ignored.

The **expression** attribute can be considered as the conditional statement for the object or the group of objects. It provides a power way to author the sophisticated scenarios. You can see the Kids Math Quiz project sample embedded with the JM-Mobile Editor to know how to use the logic **expression** attribute of an object.

Conventions:

- the **true** has the value **1**
- the **false** has the value **0**

so we can make a logic expression as like as following:

$$((\$var_1 = 3) + (\$var_2 = 5) + (\$var_3 = 1)) = 3$$

if var_1=3, var_2=5 and var_3=1 then the expression becomes:

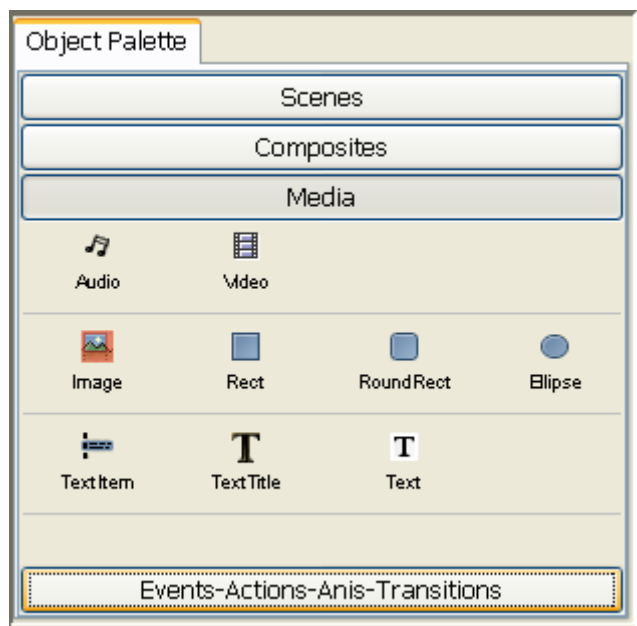
$$((\text{true}) + (\text{true}) + (\text{true})) = 3 \Leftrightarrow ((1) + (1) + (1)) = 3 \Leftrightarrow 3=3 \Rightarrow \text{true}$$

if var_3=7 then the expression becomes:

$$((\text{true}) + (\text{true}) + (\text{false})) = 3 \Leftrightarrow ((1) + (1) + (0)) = 3 \Leftrightarrow 2=3 \Rightarrow \text{false}$$

1. Object Palette

This palette is found on the top-right corner of the interface. It contains a set of types of object that author can integrate them into his/her project. Author can choose an object icon then drag it on to the *spatial view* and finally drop it at place where he/she want to integrate it.



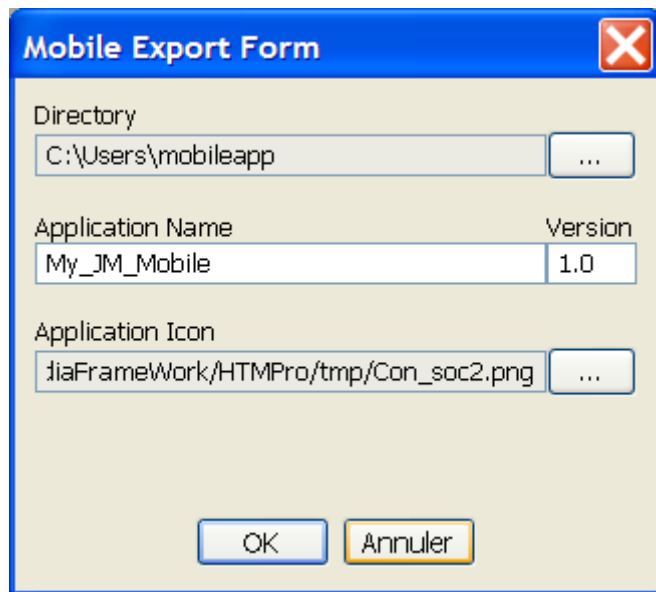
If author want to add an object inside an other object by using the *Object Palette*, he/she must firstly select the parent object on the *spatial view*, if there isn't any selected object in the spatial view, the new object will be inserted into the current *scene*.

The set of type of object is classified into four categories: *Scenes*, *Composites*, *Media* and *Events* objects (see project structure for more detail).

Exports

Export to Midlet Application

When user click on the *Export to Midlet Application* button of the toolbar, the export form following will be appeared:



The image shows a Windows-style dialog box titled "Mobile Export Form" with a red close button in the top right corner. The dialog contains three main sections: "Directory" with a text field showing "C:\Users\mobileapp" and a browse button "..."; "Application Name" and "Version" with text fields showing "My_JM_Mobile" and "1.0" respectively; and "Application Icon" with a text field showing "diaFrameWork/HTMPro/tmp/Con_soc2.png" and a browse button "...". At the bottom are "OK" and "Annuler" buttons.

Directory

Author can choose a destination directory

Application name

Author can enter a name for application

Version

Author can set a version number for application

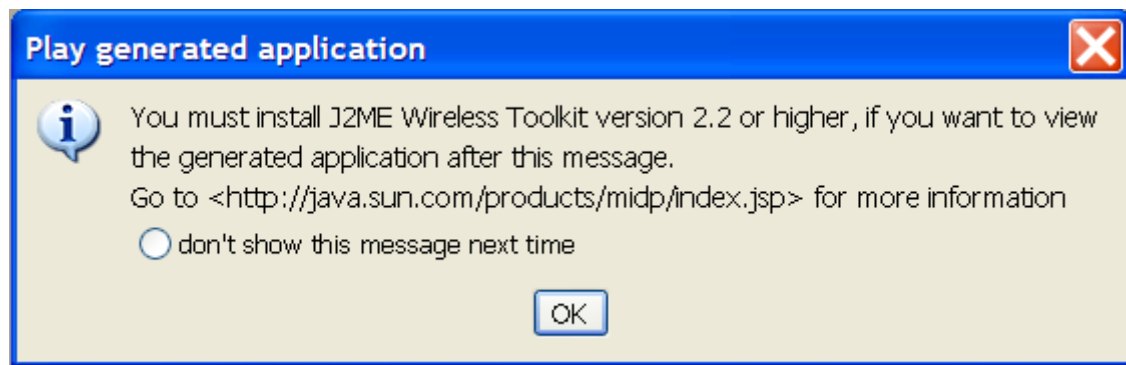
Application icon

Author can set a version number for application

When user enters all information into the form and clicks on *OK* button, the export process will be started.

All resources (images) used in the project will be packaged with the application into the jar file.

At the end of the process a message dialog following can be appeared:



It announces to user that he/she must install the J2ME Wireless Toolkit version 2.2 or higher to see the generated application.

User can choose to don't show this message next time. To show this message against, user can go to *Export -> Show playback request* menu.

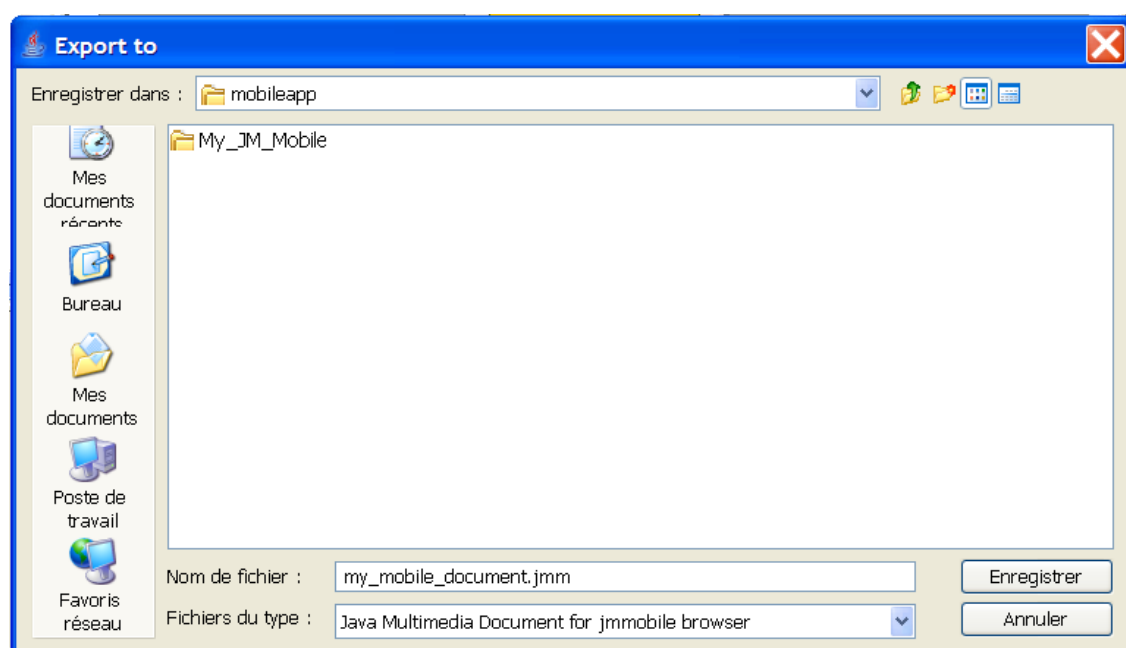
The exported application can be found at *Directory/Application Name/* containing two files:

- *Application name.jad*
- *Application name.jar*

User can then put these files to a web server to distribute the application online, or can directly transfer the jar file to a mobile-phone by using a USB or Bluetooth connection.

Export to *.jmm Document

When user clicks on the *Export to *.jmm document* button of the toolbar, the file exported specification dialog following will be appeared:



The user can choose a directory, then enter a file name, finally click on *Save* button of the dialog. The export process will be started.

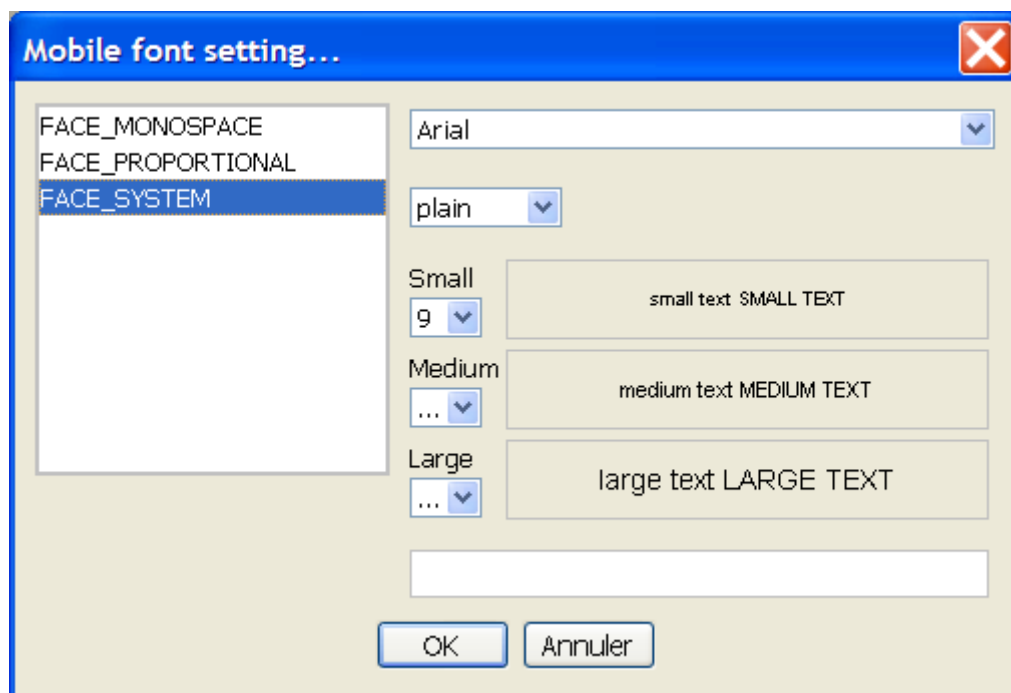
At the end of the process user can find in the directory specified the generated document *.jmm (ex. *mymobile_document.jmm*) and a *Res* directory that contains all resources used by the document.

The user can put the *.jmm document and its resource directory (*Res*) on a web server to distribute the document online.

Setting

Mobile Font Setting

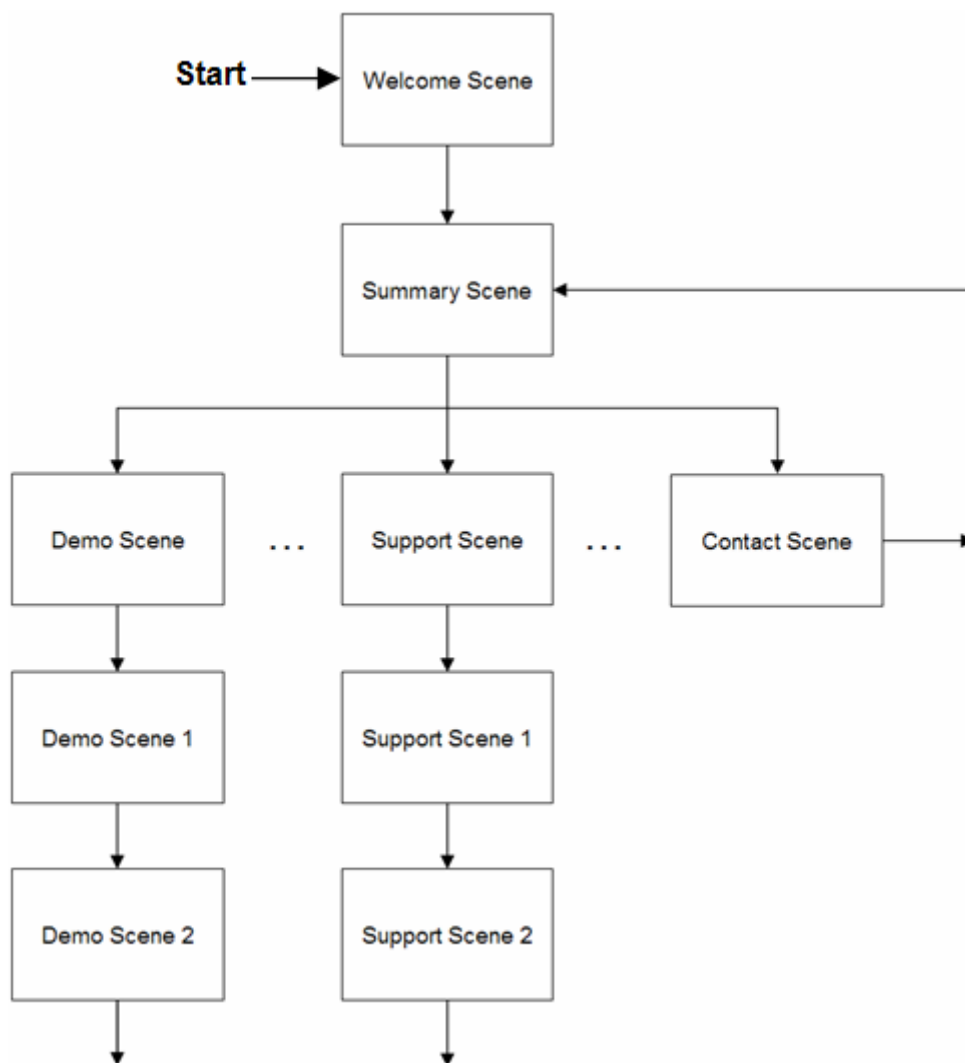
User can change the font's attributes that the authoring tool using to simulate the real fonts on the mobile-phones. He/she can go to the "*Edit -> Mobile Font Setting ...*" menu, the following form will be appeared:



The user can choose a font family (ex. the *Arial*) according to a *face* attribute (ex. the *FACE_SYSTEM*) of the mobile platform and a font size number according to each size attribute (*SMALL*, *MEDIUM*, *LARGE*) of the mobile platform.

Project Structure

The multimedia project edited with JM-Mobile Editor is composite from a set of basic media (image, text, audio, video, geometry, etc.) and composite media (menu, list of image/text, Seq, Par, Excl, etc). This composition accords with a structure as like as the structure of the theatre, i.e., it consists of a set of *scenes*. The figure below presents a scenario example of structure of a project.



May be you will have a question that why the structure of *scenes* is used in stead of the structure of *pages* or *screens*? The answer is simple: the *pages* and *screens* are referred to a static presentation, but JM-Mobile Editor allows to integrate the time dimension into the project, i.e., the JM-Mobile's presentation can be evolved in the time. So we decide to choose the term *Scene* for the structure of the JM-Mobile's project.

It is importance to note that the project structure of JM-Mobile Editor is extensible. It provides by default only the basic structures such as *generic scene*, *slideshow scene*, *menu*, *list* of image and text, *slideshow*, etc. If you need a particular structure such as *Quiz*, *Test*, *Magazine*, etc. to edit more easily your project, please contact us at support@jm-mobile.com.

XModel

You can use the XML data to create your application. The *XModel* part allows declaring the XML data that will be used in your application.

Using XML data you can:

- Structure the using data: XML is the ideal tool to organize the using data in the project; well organizing the using data makes the creation, the evolution and the management of the project easier.
- Separate the using data from the application: the using data organized in a XML data source could be daily changed or generated dynamically from a data server that doesn't request to recompile and to redeploy the application. See the NewYorkTimes sample enclosed in the JM-Mobile Editor (File->Open sample project ->NYTimes.eman).

XInstance

To declare a XML data, you insert into the XModel part a *XInstance* element.

To specify a XML data source for the *XInstance* element using its *FileName* attribute: the XML data source can be a local or online XML file or an URL of a web service that can provide a XML data stream.

Each *XInstance* element has an identity *Name*, which will be used to access into the XML data. The formula to access into a XML data represented by a *XInstance* element is following:

{*\$XPath*:document(*XInstance Name*)/*XPath*}, in some particular case such as in an expression, you can use a simpler formula

document(*XInstance Name*)/*XPath*

To access into the XML data you must use the *XPath*, but there are only a subset of XPath supported.

The XPath functions supported are:

- **count(*NodeSet*)**: to get the total number of the nodes in a nodeset.
- **substring(string, start index, count)**: the substring function such as document(data)//item[substring(@title,1,1) = 'A'] get a *nodeset* containing the *item* elements of which the title attribute is begun with 'A'. See Bosnalijek contact list sample enclosed with the JM-Mobile Editor for more detail.
- **boolean contains(string, string)**: the contains function returns true if the first argument string contains the second argument string, and otherwise returns false. By example you can use the template by XPath following document(nytimesItem)//item/title[contains(., '\$Var_content')]/.. to search the interested content from a Newyork time RSS category info. See the *newyorktimes.eman* sample enclosed with the JM-Mobile Editor for more detail.

Attention! The XML data source provided to the *XInstance* element must be the well-form XML data.

Example:

The *NewYorkTimes* sample has a *XInstance* element named *nytimesData* representing the XML file following:

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <item>
      <title>HOME</title>
      <link>http://www.nytimes.com/services/xml/rss/nyt/HomePage.xml</link>
    </item>
    <item>
      <title>BUSINESS</title>
      <link>http://www.nytimes.com/services/xml/rss/nyt/Business.xml</link>
    </item>
    <item>
      <title>EDUCATION</title>
      <link>http://www.nytimes.com/services/xml/rss/nyt/Education.xml</link>
    </item>
    ...
  </channel>
</rss>
```

You can access to the title of each item by the specification as following:

```
{XPath:document(nytimesData)//item[index(DynamicMenu_6659)]/title}
```

Where: *nytimesData* is the name of the *XInstance* element; *index(DynamicMenu_6659)* is the index function to determine the number of the current selected item of the XML menu template (*XMenuTemplate*) identified as *DynamicMenu_6659*, see the *XMenuTemplate* object part for more information about this object.

To count the total number of the item elements, you can use the specification as following:

```
count(document(nytimesData)//item)
```

memoryManager attribute

The *XInstance* element has the **memoryManager** attribute that can tell the system free or keep the XML data in the heap memory each time the application is changed to another scene.

If the XML data is only used in a scene or some few scenes, it is recommended to use the **free** value. Otherwise if the XML data is the principal data for entire application then use **keep** value for the **memoryManager** attribute, the XML data will be always on the heap

memory, the translation between the screens could be more fluid. However be really attention when using the **keep** value.

The **free** is default value.

VARs

The variables provide the ways to keep and to modify the data and/or the states of the users.

Var element

To define a variable you can choose to insert a *Var* element into the VARs part of your project. By now JM-Mobile only provides two primitives variable types: STRING and INTEGER.

The value of a *Var* can be persistent. If you want the value of a *Var* is saved between the application's uses, you can choose the value of the attribute *isPersistent* being *true*.

Modify the value of a variable

On an event occurred (begin, end, key pressed, etc.) author can choose the *VarModifyAction* behavior to modify the value of a variable. With an INTEGER variable author can choose to *set*, *subtract* or *plus* a value to the variable; with an STRING variable author can choose to *concat* or *replace* a new string to the variable.

The value used to modify a variable can be an expression. By example, on the *ItemSlideshowView* scene of the *NewYorkTimes* sample, when the *Right* key is pressed the value of the *Variable_5821* variable will be set to the value of the expression below:

```
((count(document(nytimesItem)//item) + $Variable_5821 -2) %  
count(document(nytimesItem)//item))+1
```

Variable usage

Using of variable make the presentation of application more dynamical. In general, to use a variable we put a \$ at the beginning of the variable's name (e.g. \$myScore). In some case such as using the variable in the text content, you must put it inside a pair of { }, e.g., { \$myScore }

Using variable in the text content

By example:

We have a variable called *myScore* that keeps a score of user. In the end of application we want to show this score, we can create a Text object and edit its text content as following:

"My score { \$myScore }"

*Using variable in the **FileName** attribute of an object*

This application of variable allows adapting the presentation of an object such as image corresponding with the situation of the user's interaction.

By example:

We have a string variable called *stateOfMind* that can take a “happy”, “sad”, “lovesick”, etc. then we want to show an image corresponding with a state of user in the application, we can create an image object with its *FileName* attribute specified as following:

`“.../res/images/{$stateOfMind}.png”`

AdditionalRes

If you use the dynamic media, of which the resources are only determinate at the running time in according to the user's state/data, you must add a set of the potential resources available for these dynamic objects into the *AdditionalRes* part.

By example: a dynamic image object of which the *FileName* attribute is “*res/images/{\$stateOfMind}.png*”. The *stateOfMind* variable can have one of the values [“happy”, “sad”, “lovesick”]. So the resource image of this image object can be one of the following images:

- “res/images/**happy**.png”
- “res/images/**sad**.png”
- “res/images/**lovesick**.png”

These images must be declared by adding them into the *AdditionalRes* part.

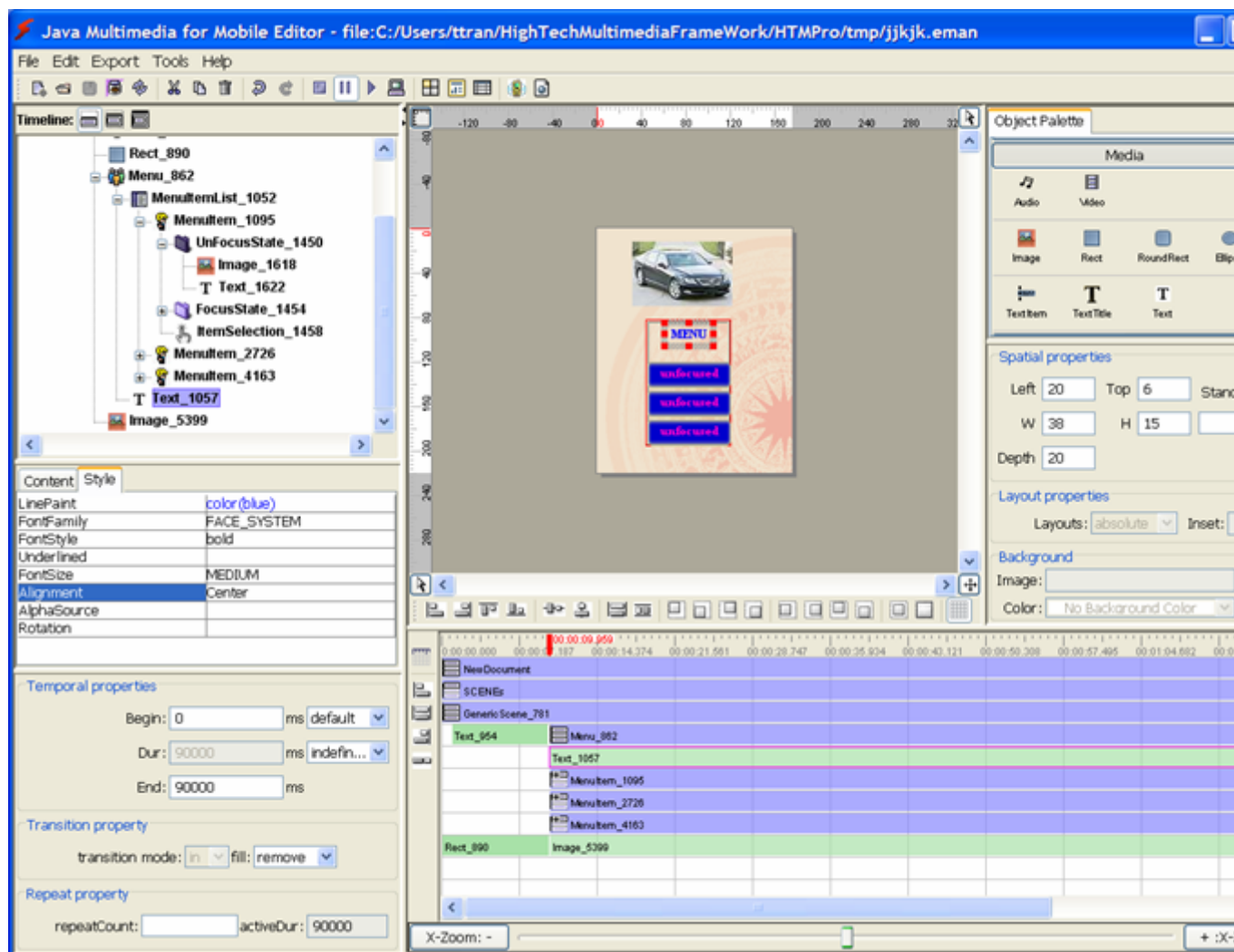
The *AddResGroup* element of the *AdditionalRes* part allows you to add in one time many resources in a same folder.

Media

Media is the basic component in the structure of JM-Mobile project. It consists of *image*, *text*, *geometry*, *audio* and *video*. Author can integrate these media type into his/her project to make the multimedia presentations.

Text

The *Text* media element of the JM-Mobile Editor allows author to integrate a text object in to his/her project and then to edit this text object. The figure below presents a instance of the interface where author is selecting and editing a text object:



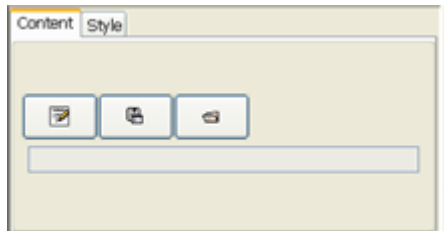
Spatial and temporal edition

Author can use the Spatial view and Spatial Attribute view to modify the spatial position and dimension of the selected text object.

Text content edition

When author double clicks on the text object region on the spatial view, a text edition window will appear allowing author to edit the text content.

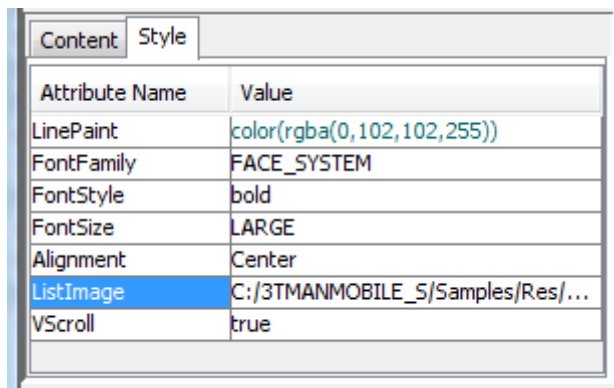
If author want to integrate a text from a file, he/she can choose the *content* tablet on the *Content, Style and Attributes view*. The *content* tablet of the text media looks like following:



thanks to the toolbar author can choose to open, save or save as a text file.

Text style edition

To edit the styles of the text object, author can choose the *style* tablet on the *Content, Style and Attributes view*. The *style* tablet of the text media looks like following:



Attribute Name	Value
LinePaint	color(rgba(0,102,102,255))
FontFamily	FACE_SYSTEM
FontStyle	bold
FontSize	LARGE
Alignment	Center
ListImage	C:/3TMANMOBILE_S/Samples/Res/...
VScroll	true

Authors can add an image at begin of text, use the ListImage style attribute of the text object (see the picture below)

Text With VScroll and ListImage at Begin of the text



Jmm documents
exported from
JMMobile editor
can be put in any
Web server to
distribute, then any javaenable
cellphone installed our JMM
Browser can be download and
playback them. That doesn't
require any particular web

The text can scroll by specifying the VScroll attribute of the text style equal **true**

- the scroll bar is only displayed when the text's size is super than the text's area;
- when the scroll bar is displayed, we can use key NUMBER2 and key NUMBER8 to scroll UP and DOWN the scroll bar.
- see the picture above

BitmapText

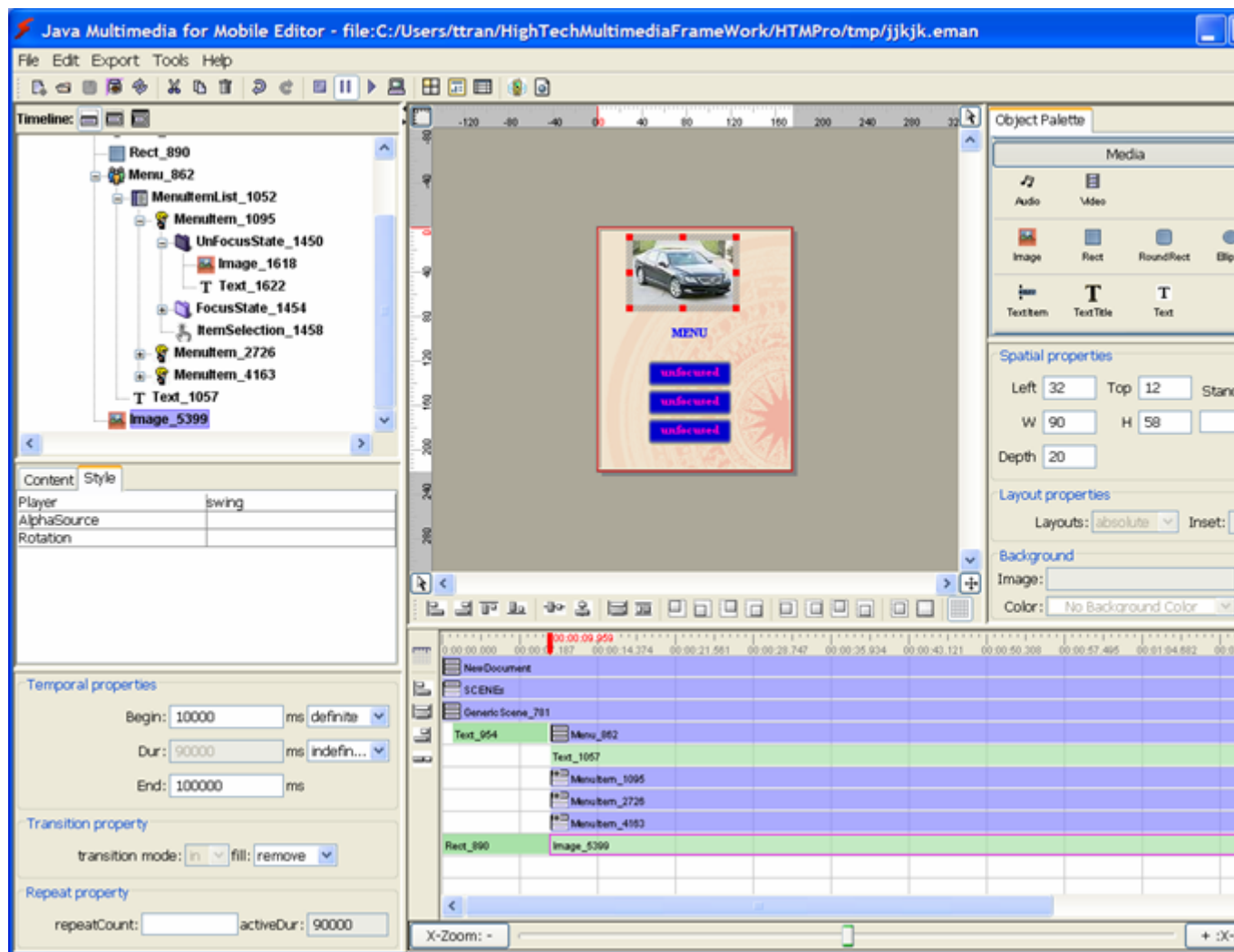
The BitmapTextIcon object allows using the bitmap fonts. In fact, The java mobile font system is implemented very different on different java-enable cell-phones. Using the bitmap font will help you manage the text displaying problems and make your applications to be displayed correctly on different cell-phones. Your application so becomes more portable

However you must be careful when using the bitmap font! Don't use too many bitmap fonts that will decrease the performance of your application.

Integrate the Bitmap Font Generator tool that can help you create easily the bitmap font for jm-mobile editor. This tool be accessed from menu tools->Bitmap Font Generator...

Image

The *Image* media element of the JM-Mobile Editor allows author to integrate an image object in to his/her project. The figure below presents a instance of the interface where author is selecting and editing a image object:



Spatial and temporal edition

Author can use the Spatial view and Spatial Attribute view to modify the spatial position and dimension of the selected text object.

Image content edition

If author want to change a image file, he/she can choose the *content* tablet on the *Content, Style and Attributes view*.

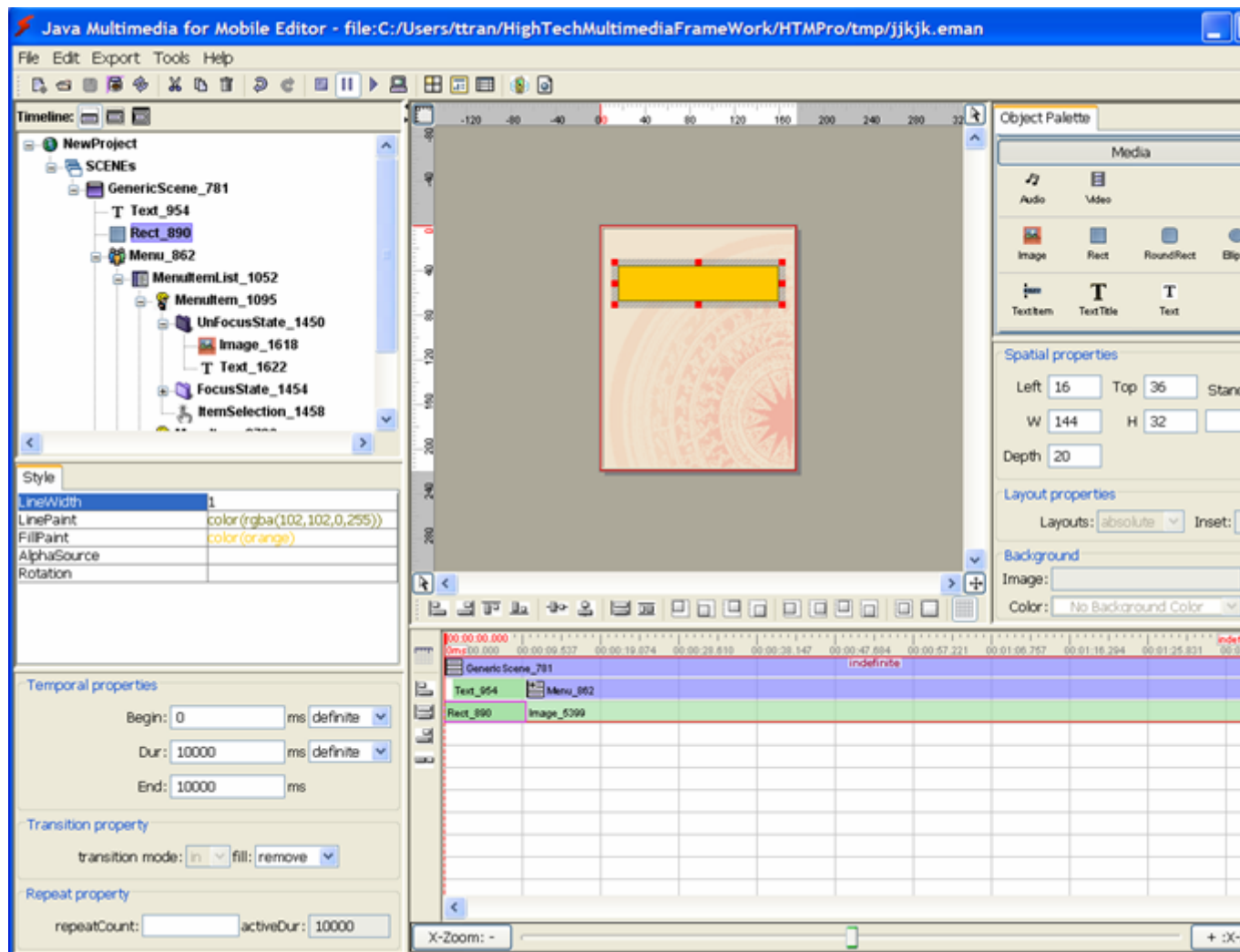
Attention! it is strongly recommended that author should use only the small images.

Author can choose the *AniImageIcon* object to integrate into project the gif animation images.

The Image object can use the image source from the Internet, i.e. `src="http://www..."`

Geometry

The *Geometry* media element of the JM-Mobile Editor allows author to integrate a geometry (*rectangle*, *round-rectangle* and *eclipse*) object in to his/her project. The figure below presents a instance of the interface where author is selecting and editing a rectangle object:



Spatial and temporal edition

Author can use the Spatial view and Spatial Attribute view to modify the spatial position and dimension of the selected text object.

Geometry style edition

Author can modify the style of geometry object by using the *style* tablet.

Attention! some style attributes are not supported when the application is playback on the real mobile phone such as *AlphaSource* and *Rotation* attribute. The *LineWidth* attribute is only supported 0 and 1 value.

Audio

The **Audio** media element of the JM-Mobile Editor allows to author to integrate an audio (*wav*, *mp3*) object in to his/her project. Difference with the text, image and geometry are the static media, the audio and video media are the dynamic media which have themselves an intrinsic duration time.

In addition, though audio and video are rich media that can make your project more attractive and more exciting, they are also the heavy media that require to consume much resource. Therefore if you want your application can be quickly downloaded and can playback on as many as possible mobile-phone, you should avoid using audio and video media.

If your project must use the audio and video media, you should choose to use the audio and video in mode streaming, not in the local mode.

Finally, to work with audio and video media in the JM-Mobile Editor you must install JMF (Java Multimedia Framework).

Video

The **Video** media element of the JM-Mobile Editor allows author to integrate a video object in to his/her project. Difference with the text, image and geometry are the static media, the audio and video media are the dynamic media which have themselves an intrinsic duration time.

In addition, though audio and video are rich media that can make your project more attractive and more exciting, they are also the heavy media that require to consume much resource. Therefore if you want your application can be quickly downloaded and can playback on as many as possible mobile-phone, you should avoid using audio and video media.

If your project must use the audio and video media, you should choose to use the audio and video in mode streaming, not in the local mode.

Finally, to work with audio and video media in the JM-Mobile Editor you must install JMF (Java Multimedia Framework).

Attention:

- Many the real devices today just supports only the *.3gp video
- The WTK Simulator can't playback the *.3gp videos, however these video will be correctly played on the real device that supports multimedia for java (mmapi)
- We can also use the video from distance (internet). However the editor can't playback the *.3gp from distance, but the exported application can play correctly the video on the real devices.
- The editor can playback the *.mpg from distance and then the simulator can also playback it, but on the real devices only the *.3gp are supported
- You can try to edit the video object with the following distance videos to see more
 - <http://java.sun.com/products/java-media/mma/media/test-mpeg.mpg>
 - <http://video.cityneo.com/videoperso/61.3gp>

Event and Behavior

The Event and behavior are the main interactive components of the multimedia presentations. The JM-Mobile Editor provides the **KeyAccess** object to definite an event that author want to capture and a behavior following the event.

KeyAccess

The **KeyAccess** object can be added into a media object such as image, text, geometry, etc. It allows to definite an *accessible key* to fire an *action behaviour* during the playing time of the media object. The multiple key access events on an object are supported.

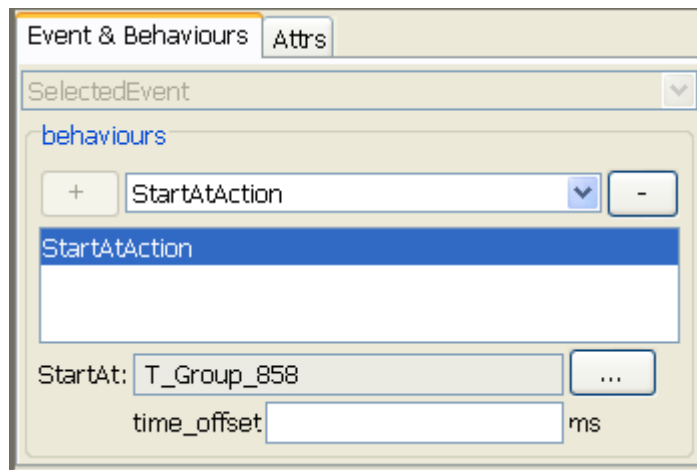
The Key_Value attribute

Author can specify a key access value through this attribute. It can take one of the following values:

FIRE
LEFT
RIGHT
DOWN
UP
KEY_NUM0
KEY_NUM1
KEY_NUM2
KEY_NUM3
KEY_NUM4
KEY_NUM5
KEY_NUM6
KEY_NUM7
KEY_NUM8
KEY_NUM9
KEY_STAR
KEY_POUND

The Behaviours specifications

The figure below presents the interface for specifying an action:



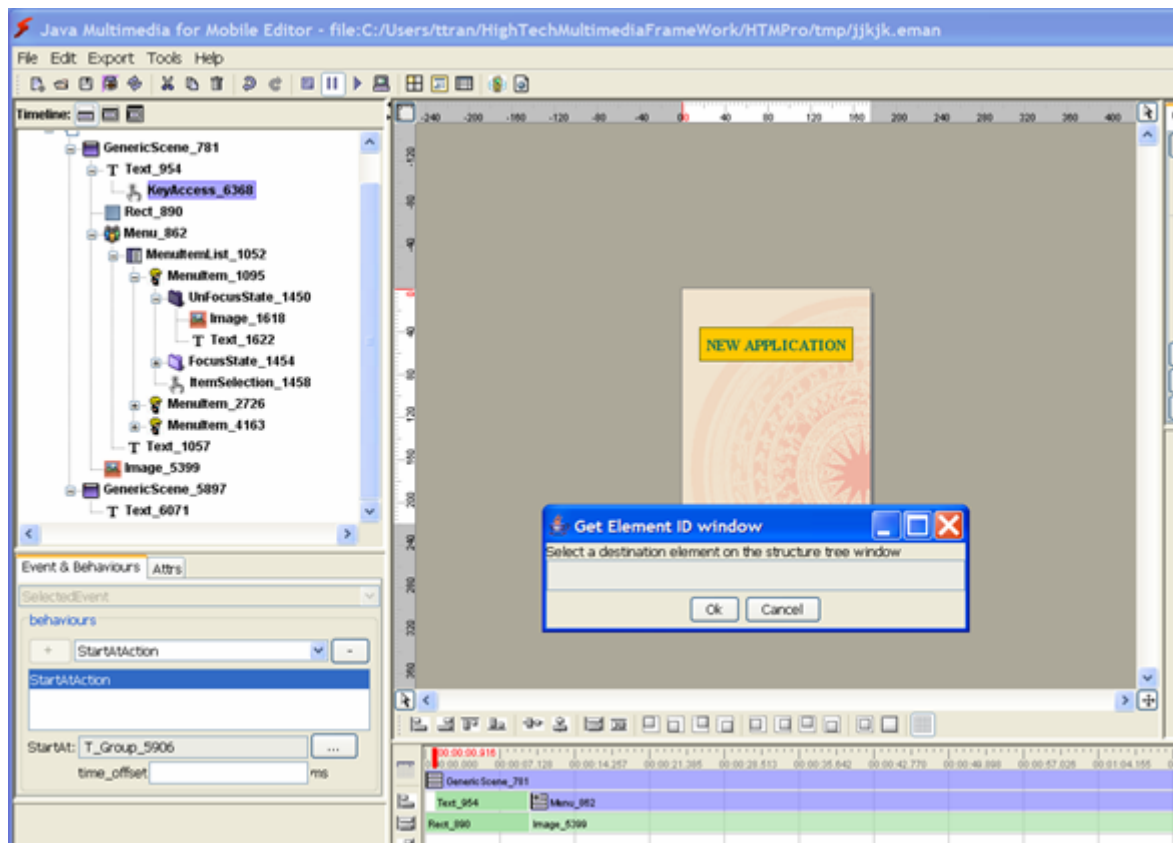
A Combo box allows author to choose an action. When an action is chosen, the author can press button "+" to add the action into the list of actions (in this version we allow to add only one action into the list). The author can use the "-" button to delete the action.

According to each chosen action the interface provides a particular interface to specify the attributes of that action.

The JM-Mobile Editor provides a set of actions that author can specify to be fired following an key typed event.

[StartAtAction](#)

This action allows the presentation to jump over time to the beginning of an object. To specify the *StartAt* object, author clicks on button "...", an object selection window will be showed as below:



Author uses the *structure view* to select the *StartAt* object, then clicks button "Ok" to take a count the selected object. He/she can choose the button "Cancel" to show off the window and do not take a count the selected object.

This action is often used to jump between the deferent scenes of a multimedia presentation.

StartElementAction

This action is simply playing a specified object, there isn't a jump over time. This action is often used to start an object of which the *Begin* attribute is *indefinite*.

EndElementAction

This action allows to forcedly stop an specified object.

PhoneCallAction

This action allows to activate the *Phonecall* of the mobile device with a specified calling number.

HyperlinkAction

The *HyperlinkAction* behaviour allows to launch the default Internet browser of the mobile-phones to:

- browse a page web
- download an URL resource (image, video, audio, etc). The Video on Demand sample application on the next side provides a menu list that the users can select one item of the menu to download a video through an URL address from <http://video.cityneo.com/videoperso/xxx.3gp>
- download and install the jar applications
- send the data result of the user interaction (e.g. result of a test, a user note, etc.) to a Web server
- etc.

The *HyperlinkAction* behaviour can be used to make a hyperlink to the other JMM documents exported by JM-Mobile. This capacity of hyperlink between the JM-Mobile documents is really powerful and interesting. It allows to create a JM-Mobile presentation network on which the user can navigate as like as the web navigation (try the hyperlink project sample).

Composite

Composite is a composition of media and other composites according to a structure particular. For instant, the *Image List* is a composite structure that allows to integrate easily together a set of images.

There are more sophisticated composites that can be integrated with the *event and behavior* objects to be enable the interactions with the composites, such as the *Slideshow* and the *Menu* composites.

By default the authoring tool is only integrated the generic composites such as *List*, *SlideShow*, *Menu*, etc. If your project needs more particular structure composites, please contact us at support@jm-mobile.com .

Temporal Group

The temporal layout group consists of the *Par*, *Seq* and *Excl* groups that allow to integrate easily a set of media and composite objects in the time. This temporal group is adopted from the timing and synchronized model of SMIL, you can see this model at www.w3c.org for more detail.

It is importance to note that this temporal group and the spatial composite group are the most basic composites, that all other composites are built rely on.

Par Group

The *ParGroup*, short for "parallel", defines a simple time grouping in which multiple media and/or other composites can playback at the same time.

When an object is added into this *ParGroup* its temporal attributes are set by default as following:

- *Begin* = 0
- *Dur* = *indefinite*

This temporal group is adopted from the *par* container of SMIL timing and synchronized model, you can see this model at www.w3c.org for more detail.

Seq Group

A *SeqGroup* defines a sequence of media/composite objects in which objects play one after the other.

When an object is added into this *SeqGroup* its temporal attributes are set by default as following:

- *Begin* = *End* value of the object before, if the added object is the first child, its *Begin* is set to 0
- *Dur* = *Dur* value of the object before, if the added object is the first child, its *Dur* is set to 10000ms

This temporal group is adopted from the *seq* container of SMIL timing and synchronized model, you can see this model at www.w3c.org for more detail.

Excl Group

The *ExclGroup*, short for "exclusive", defines a time grouping in which only one media/composite object can playback at a time. If any object begins playing while another is already playing, the object that was playing is stopped.

When an object is added into this *ExclGroup* its temporal attributes are set by default as following:

- *Begin* = *indefinite*
- *Dur* = *indefinite*

So you must change the default *indefinite* value of begin attribute, if you want the object playing.

The *ExclGroup* composite provides an integrating structure with which author can compose the very interesting presentations. In the *event and behavior* part you can see an interesting sample compositions of this *ExclGroup* composite with the *event and behavior*.

This temporal group is adopted from the *excl* container of SMIL timing and synchronized model, you can see this model at www.w3c.org for more detail.

Spatial Layout Group

The spatial layout group consists of the *Absolute* and *List* groups that allow to integrate easily a set of media and composite objects in 2D space.

It is importance to note that this spatial layout group and the temporal composite group are the most basic composites, that all other composites are built rely on.

Absolute Layout

The absolute layout provides a spatial container that allows author to insert into it any media or composite objects, to move and to place these objects at any position in the spatial container.

List Layout

The list layout provides a spatial container that allows author to insert into it any media or composite objects following a vertical or horizontal layout.

Image List

The list image is a high level composite media that provides a sample way to integrate a set of images objects. There are four type of image list: *vertical*, *horizontal*, *absolute* and *sequential*

Vertical image list

This type of image list is built rely on a *ParGroup* and a *VerticalListLayout*. It allows to grouping the image objects in the manner that the images can be showed in a same time and they are aligned vertically.

Horizontal image list

This type of image list is built rely on a *ParGroup* and a *HorizontalListLayout*. It allows to grouping the image objects in the manner that the images can be showed in a same time and they are aligned horizontally.

Absolute image list

This type of image list is built rely on a *ParGroup* and a *AbsoluteListLayout*. It allows to grouping the image objects in the manner that the images can be showed in a same time and they haven't not any spatial constraint.

Sequential image list

This type of image list is built rely on a *SeqGroup* and a *AbsoluteListLayout*. It allows to grouping the image objects in the manner that the images are showed one after the other and they haven't not any spatial constraint.

Text List

The list text is a high level composite media that provides a sample way to integrate a set of text objects. There are four type of text list: *vertical*, *horizontal*, *absolute* and *sequential*

Vertical text list

This type of text list is built rely on a *ParGroup* and a *VerticalListLayout*. It allows to grouping the text objects in the manner that the texts can be showed in a same time and they are aligned vertically.

Horizontal text list

This type of text list is built rely on a *ParGroup* and a *HorizontalListLayout*. It allows to grouping the text objects in the manner that the texts can be showed in a same time and they are aligned horizontally.

Absolute text list

This type of text list is built rely on a *ParGroup* and a *AbsoluteListLayout*. It allows to grouping the text objects in the manner that the texts can be showed in a same time and they haven't not any spatial constraint.

Sequential text list

This type of text list is built rely on a *SeqGroup* and a *AbsoluteListLayout*. It allows to grouping the text objects in the manner that the texts are showed one after the other and they haven't any spatial constraint.

Slideshow

Slideshow is a structrue grouping a list of slides in which the *slides* are presented one after others.

To control the navigation within a *Slideshow* composite, going to the next slide and reverting to the previous slide, author can use the *Navigation_buttons* attribute of this composite. The attribute can get one of two values: *"VK_UP;VK_DOWN"* or *"VK_LEFT;VK_RIGHT"*.

- *"VK_UP;VK_DOWN"* value allows to use *Up* and *Down* keys two navigate within the *Slideshow* composite.
- *"VK_LEFT;VK_RIGHT"* value allows to use *Left* and *Right* keys two navigate within the *Slideshow* composite.

Menu

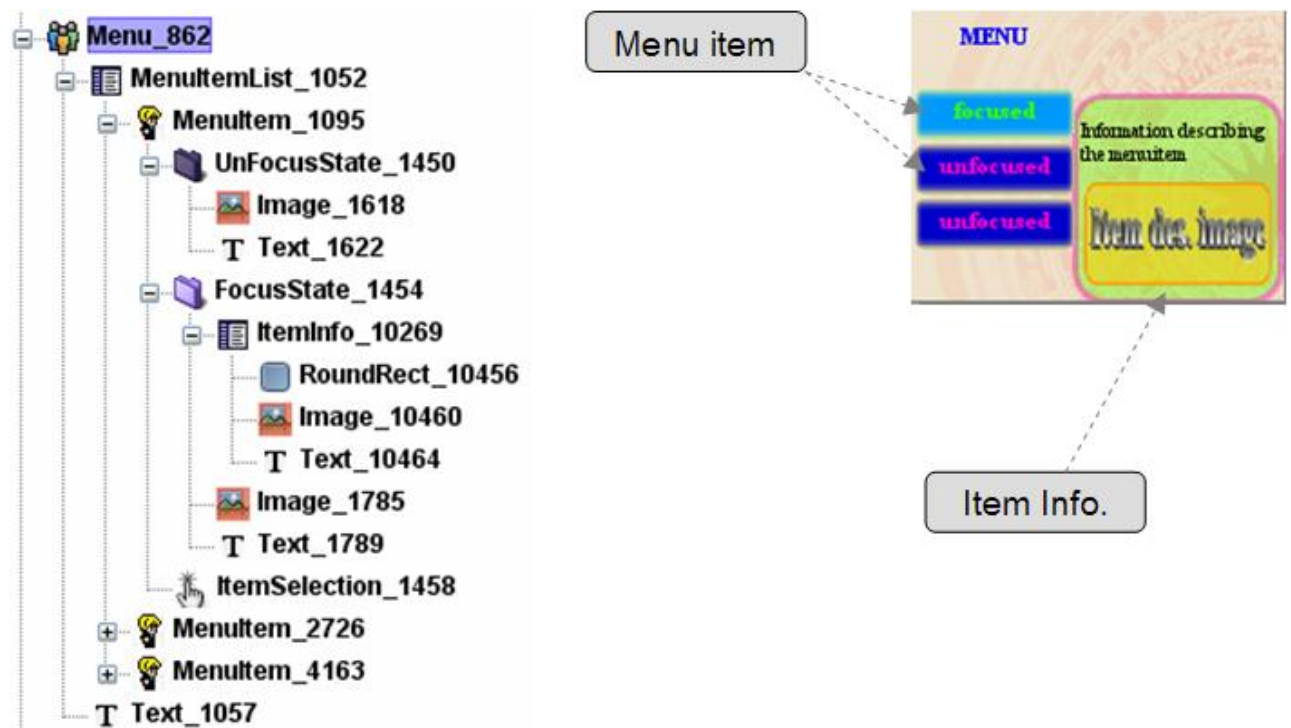
Menu is the sophisticated composite provided by default in the distributed JM-Mobile Editor. There are actual two menu types: *FlowLayoutMenu* and *AbsLayoutMenu*

FlowLayoutMenu

The *FlowLayoutMenu* automatically organizes its *MenuItem* elements in the vertical or horizontal direction depending on the flow layout being vertical or horizontal.

The navigation keys between the items of this menu type are also implicitly specified: UP and DOWN arrow keys for the vertical flow layout menu; LEFT and RIGHT arrow keys for the vertical flow layout menu.

The structure and presentation of *FlowLayoutMenu* look like as the figure below:



The left part is the structure of a *Menu* presented in the *Structure View*, and the right part is one screen shot of the menu presented on the *Spatial Layout View*.

MenuItemList

The *MenuItemList* is a sub structure of the *Menu*. It allows to group the menu items of a menu, and organize them in the space (*vertical*, *horizontal* or *absolute*).

MenuItem

The *MenuItem* is a structure allowing to manage two states (*UnfocusState* and *FocusState*) of a menu item and the selection event (*ItemSelection*) of the menu item. It is built rely on the *Exclusive* temporal structure that allows only an item's state (*UnfocusState* or *FocusState*) presenting at a time.

The *ItemSelection* is an *Event and Behavior* structure that allows to specify a fired action when the focusing item is selected (see the *Event and Behavior* part for more information).

State of MenuItem

The states of menu item (*UnfocusState* and *FocusState*) are the structures allowing to grouping media/composite objects to create a presentation of a menu item at a state.

ItemInfo

In addition, the structure of *FocusState* allows to integrate one more *ItemInfo* structure. The *ItemInfo* structure allows author to present more information about the item in focusing (see the figure).

Inside the *ItemInfo* structure you can create a *SlideShow* structure, such the composition allows authoring the sophisticated scenarios. You can open the *Nokia_Catalogue* sample to see how such composition can be done.

AbsLayoutMenu

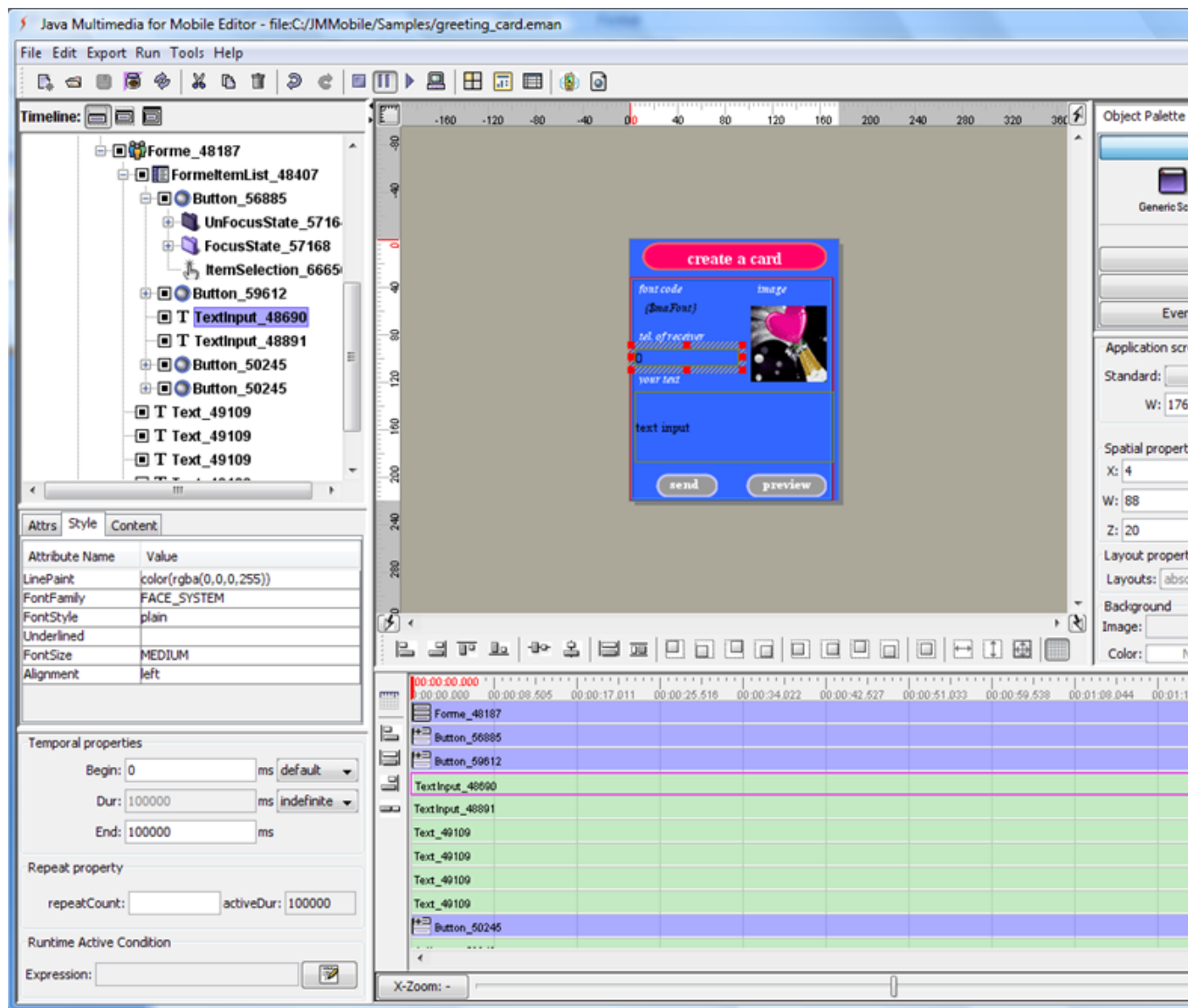
Different from the *FlowLayoutMenu*, the location of the menu items of the absolute layout menu isn't constrained, it can be freely located.

The navigation between the menu items of the absolute layout menu must be explicitly specified. The menu item of the absolute layout menu (*AbsMenuItem*) possesses a set of the attributes: [left, right, up, down] to specify the next menu items to navigate if the corresponding arrow keys are pressed.

Form

Form is the composite element that allows to author the navigable structures in which the children components aren't the same types (Menu is the composite navigable structure in which the children components are identical, it's the Menu Item).

The figure below represents a form structure in the *greeting_card* project sample:



The children components of the *Form* can be the *TextInput* and/or the *Button*. These children component can be located freely inside the *Form* component. The way to define the navigation between the children components of the *Form* is as like as the navigation specification among the absolute menu items (see the *AbsLayoutMenu*).

As like as the absolute menu items, the *TextInput* and/or the *Button* possess the set of the attributes [left, right, up, down] to define the next item to navigate if the corresponding arrow keys are pressed.

TextInput

The *TextInput* allow to define a way to enter the user's data. The user's data can be an INTEGER, a STRING or a PASSWORD. The value of the *TextInput* can be used by referencing to the name of the *TextInput*. The way to use the *TextInput*'s value is as like as the way to use the variable. See the variable for more information.

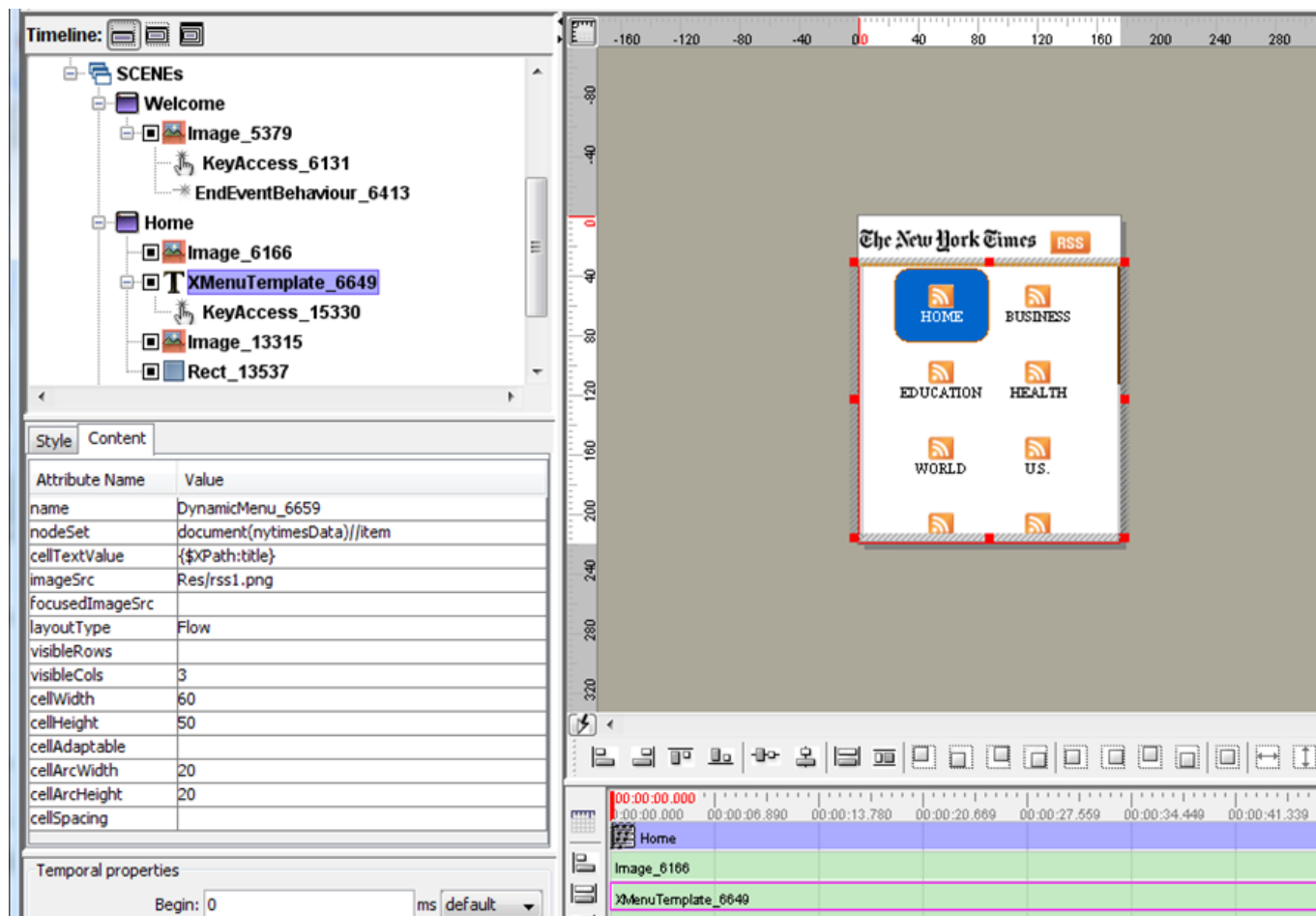
Button

Button is a focusable structure that allows specifying the rendering of the component in two states: *Focused* and *Unfocused* state.

XMenuTemplate

XMenuTemplate allows creating the menus with the XML data. In other way, the **XMenuTemplate** defines the template to present the XML data and to navigate on the XML data.

The figure below represents a **XMenuTemplate** element in the *NewYorkTime* project sample:



The XML data referenced by this XMenuTemplate element is following:

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <item>
      <title>HOME</title>
      <link>http://www.nytimes.com/services/xml/rss/nyt/HomePage.xml</link>
    </item>
    <item>
      <title>BUSINESS</title>
```

```

    <link>http://www.nytimes.com/services/xml/rss/nyt/Business.xml</link>
  </item>
  <item>
    <title>EDUCATION</title>
    <link>http://www.nytimes.com/services/xml/rss/nyt/Education.xml</link>
  </item>
  ...
</channel>
</rss>

```

To define a *XMenuTemplate* element, you must specify a set of the attributes which is devised into two attribute groups: the *Content Attributes* and the *Style Attributes*

Content Attributes

name

Name attribute allows specifying the identification of the *XMenuTemplate*. The *XMenuTemplate*'s name will be used to reference to the *XMenuTemplate*.

nodeSet

nodeSet attribute allows specifying a set of XML data nodes which will provides the data for the XML menu template. Each XML data node provides the data for the menu item corresponding.

To specify a set of the XML data nodes, you must reference to the XML data represented by a *XInstance* element and then define a XPath to get a set of XML data nodes in this XML data. The example below is the specification of the *nodesSet* attribute of the *XMenuTemplate* in the ewYorkTimes project sample:

```
document(nytimesData)//item
```

Where:

nytimesData is the name of a *XInstance* in the *NewYorkTimes* project sample; *//item* is the XPath which allows to get a set of the *item* nodes in the *nytimesData* XML data.

cellTextValue

cellTextValue attribute allows to specify the text data for each menu item. You can use the static text and the dynamic text to specify this attribute. By example "*item title: {XPath:title}*". The static text "*item title:* " will be rendered the same in all menu items; the dynamic text "*{XPath:title}*" will be rendered differently on each menu item.

The dynamic text in this case is a XPath to a XML data of a XML node in the set of XML nodes specified in the *nodeSet* attribute, i.e. the context of XPath in this specification is the XML data nodes defined in the *nodeSet* attribute.

The example below represents the specification of the *cellTextValue* attribute of the *XMenuTemplate* in the NewYorkTime project sample, the context XML node is the *item*:

{XPath:title}

That means using the *title* data of the *item* XML node for the menu item corresponding

imageSrc and focusedImageSrc

imageSrc and *focusedImageSrc* attributes allow to specify an image for rendering each menu item in the unfocused state and in the focused state.

You can use a static image specification or a dynamic image specification.

The static image specification, by example *Res/rss1.png*, will render a same image on all menu item.

If you want to render each menu item with a corresponding image you must use the dynamic image specification, suppose that the item node has also the image source data as following:

```
<item>
  <title>BUSINESS</title>
  <link>http://www.nytimes.com/services/xml/rss/nyt/Business.xml</link>
  <img>businessIcon</img>
  <focusedImg>focusedBusinessIcon</focusedImg>
</item>
```

You can then define the dynamic image specification as following:

Res/{XPath:img}.png

layoutType

layoutType attribute allows to define how the menu items will be organized. There are four layout types:

- **Vertical**: the menu items will be organized as a vertical list of items
- **Horizontal**: the menu items will be organized as a horizontal list of items
- **Table**: the menu items will be organized as a table of which the row and the column are defined in the *visibleRows* and *visibleCols*
- **Flow**: the menu items will be organized continuously one item after another item. The menu items will be wrapped automatically depending on the width of the *XMenuTemplate* element

visibleRows and visibleCols

visibleRows and *visibleCols* attributes allow to define the row and the column of the table, in the case the layout type is the **Table**, in other cases these attributes are ignored.

cellWidth and cellHeight

cellWidth and *cellHeight* attributes allow to define the dimension of each menu item.

cellArcWidth and cellArcHeight

cellArcWidth and *cellArcHeight* attributes allow to define the menu items as the round rectangles.

cellAdaptable

cellAdaptable attribute allows to define a strategy for adapting the size of each menu item when the application adapts to a screen size of a mobile-phone.

cellSpacing

cellSpacing attribute allows to define a space between the menu items.

Style Attributes

The style attributes allow to define the presentation properties of the *XMenuTemplate*, such as the color, the alignment, font style, font size, etc.

Index Function

The Index function allows getting the number of the current selected item of a *XMenuTemplate* at the runtime. The usage syntax of the function is defined as following

`index(XMenuTemplate's name)`

This index function is very useful, it allows getting the data of the XML node selected by a *XMenuTemplatye*

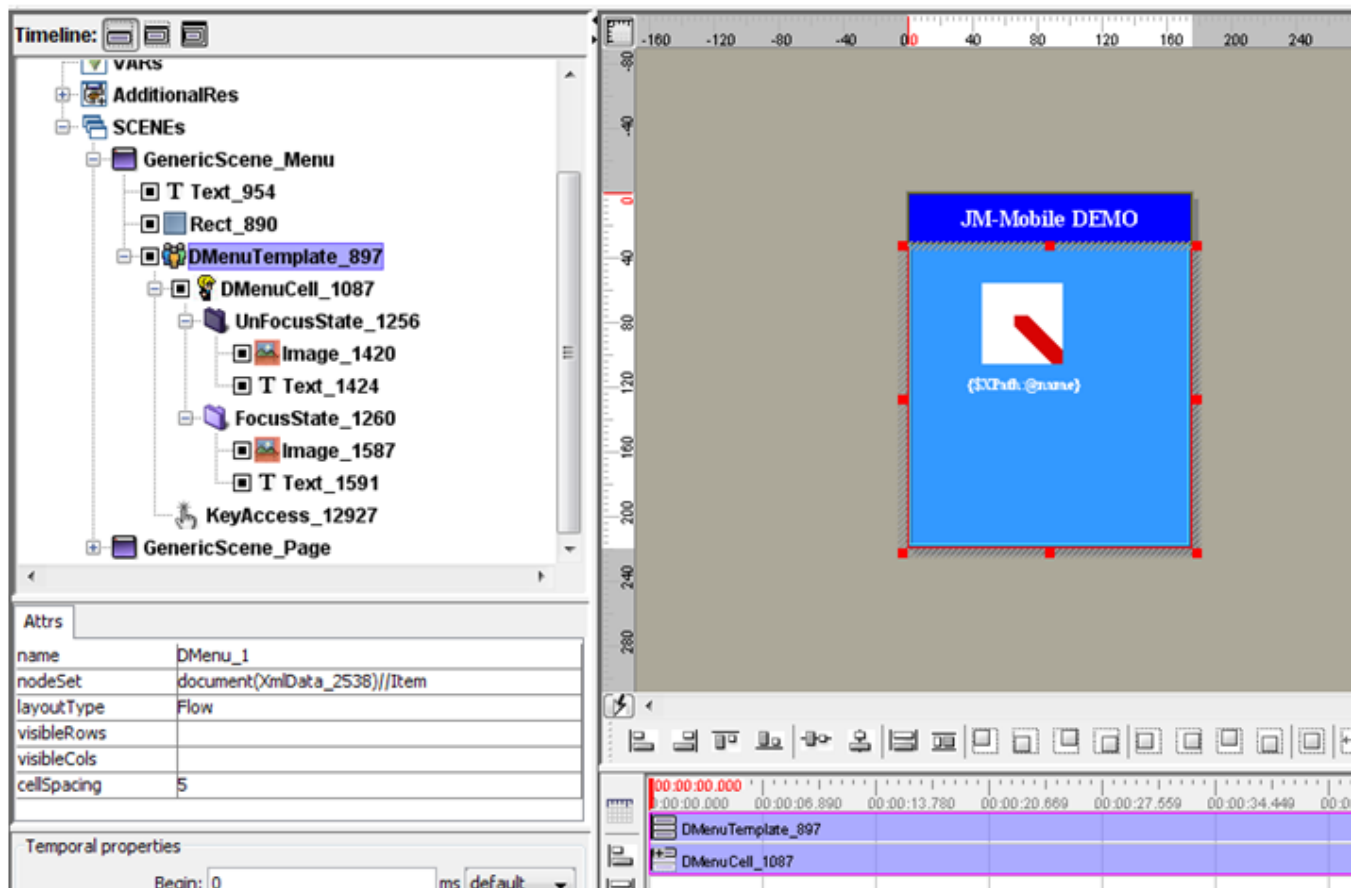
By example: to show the title of the selected item of the XMenuTemplate named *nytimesData* you can define a text object of which the text content is defined as following:

`{XPath:document(nytimesData)//item[index(DynamicMenu_6659)]/title}`

DMenutemplate

DMenuTemplate also allows creating the menus with the XML data as like as the *XMenuTample*. However the *DMenuTemplate* is much more power than the *XMenuTample* one. It allows freely and easily authoring the menu item template: the unfocused state and the focused state can be designed properly; the layout of the objects in the menu item isn't constrained, they can be located freely. In brief, with the *DMenuTemplate* you can create the much more sophisticated dynamic menus.

The figure below represents a *DMenutemplate* element in the *dynamicmenu* project sample:



The XML data referenced by this DMenutemplate element is following:

```
<?xml version="1.0" encoding="Windows-1252" ?>
<jmmobile>
  <dynamicMenu>
    <Item name = "Animations" id = "0">
      <normal>Animations_unsel.png</normal>
      <hover>Animations_sel.png</hover>
    </Item>
    <Item name = "Buttons" id = "1">
      <normal>Buttons_unsel.png</normal>
      <hover>Buttons_sel.png</hover>
    </Item>
    <Item name = "Dialogs" id = "2">
      <normal>Dialogs_unsel.png</normal>
      <hover>Dialogs_sel.png
    </Item>
    <Item name = "Fonts" id = "3">
      <normal>Fonts_unsel.png</normal>
      <hover>Fonts_sel.png</hover>
    </Item>
    ...
  </dynamicMenu>
</jmmobile>
```

As you can see in the figure above, the structure of the *DMenuTemplate* is really simple; it is even simpler than authoring a menu which has only one menu item.

Attribute Specifications

Attribute Specifications have also the similar attributes as the *XMenuTemplate* element, but they are much simpler. They require specifying only 6 attributes:

- name
- nodeSet
- layoutType
- viewRows
- viewCols
- cellSpacing

See the *XMenuTemplate* element to know how to specify these attributes.

Scene

The structure of JM-Mobile project allows author to organize his/her project following the scenes, each scene presents a part, a period, a topic/subject of the multimedia project.

There are two types of scene provided: *GenericScene* and *SlideshowScene*.

The *GenericScene* is a neutral structure that allows to edit the general multimedia presentations.

With the particular multimedia presentations such as Slideshow, Test, Quiz, Magazine, etc. if there are the particular scene structures for these particular presentations, the edition and management of these presentations will be much more easily. However, only the *SlideshowScene* structure is provided by default. If you want edit the other specific scenes please contact us at support@jm-mobile.com.

Performance hint

- "Scene" provides the ways to structure your project in different parts that will be presented exclusively. Avoid creating the big scenes which contain many rich media such as images. The big scenes will make the time to initial them to be long; in addition these scenes maybe couldn't be loaded on the weak cell-phone systems.

Performance Hint

System Requirements

Windows

- Intel Pentium 1GHz or higher
- Windows 2000 or Windows XP
- 256 MB recommended of free available system RAM

Usage recommended

- We strongly recommend saving regularly your project.
- Avoid opening many applications simultaneously with JM-Mobile Editor

Design recommended

- Don't use too many rich media such as image, audio and video for your project. If you decide to use a rich media, let select a media with the smallest size possible.
- "Scene" provides the ways to structure your project in different parts that will be presented exclusively. Avoid creating the big scenes which contain many rich media such as images. The big scenes will make the time to initial them to be long; in addition these scenes maybe couldn't be loaded on the weak cell-phone systems.

Contacting Us

The objects and the presentation structures presented in this guide are the default and common objects/structures. If you need the particular features more customized please feel free to contact us support@jm-mobile.com.

Please send support questions, feedback, suggestions and bug reports to:
support@jm-mobile.com.