iCon.text – a customizable iPad application for interactive kiosks in museums

Manual

Marco Klindt (klindt@zib.de)
Zuse Institute Berlin
http://www.zib.de/en/visual/software.html

1. Introduction

What is iCon.text? iCon.text is an iPad application, downloadable from the Apple App-Store, which can be tailored to present information and contexts about exhibits or artefacts presented in museum exhibitions. The user of the customized application will be the visitor to museum seeking more information about objects and their contexts.
The central metaphor is a postcard, that represents an object with one front side and multiple backsides. The information about such object is depicted with images. These images have to be created by the curator or designer of the exhibition.
To access these postcards one or more of several navigation views can be selected for displaying and interacting with the cards.
The application can also provide all the information in two different language versions.
This manual describes how to customize the navigation bar, the navigation views, and provide information for the postcards to be used by the application.
The application without any customization presents itself with dummy content to explore the various interaction possibilities. Several example templates can be downloaded from the support website and will be used throughout this manual to explain the configuration of the framework.
The content provider is expected to be somewhat familiar with a plist- or text editor and an image editor to create content and customization information and layout. An iPad with iTunes is obviously also needed.
All screenshots in this document are for illustration only. Your mileage may vary.

1.1 Features

- Designed for the iPad 2 in fixed landscape orientation (home button right) and fixed resolution of 1024x768 pixels. It also works on iPad and the new iPad, although not taking advantage of the retina display resolution.

These Features are customizable and will be described in more detail during the course of this manual:

- Unlimited number of *postcards* representing information about objects:
    - One front side to represent the object in the navigation views
    - Multiple backsides providing information about the object
    - Linking possibilities to link one backside to the backsides of another relevant card. Linking to a card inactivates links on the second card in order not to confuse the end user or generate infinite linking cycles
- A *navigation bar* in the bottom for accessing the navigation views where the postcards are displayed.

- A interactive animated pile of postcards (*pile view*) that provides a playful introduction to the application, user interface, and the central metaphor. This view is mandatory and should not be removed.
- A zoomable mosaic layout with all the postcards (*mosaic view*) which may also enable the visitor to view the front image of the cards in fullscreen resolution.
- A view where cards can be grouped in categories to access objects which might belong to a common theme or share similar attributes (*categories view).*
- A two level *context view* which enables the visitor to approach different contexts and the postcards therein from an overview image.
- A zoomable image with cards that can be used a representation of a floor plan or map much like google maps (*plan view).*

1.2 Contents of the support package

The support package contains a copy of this manual and a zipped archive with all the files needed to customize iCon.text with the example configuration and design. You can use it as a starting point for your own customization. The creation and explanation of this example is described in the following section.

2. Cusrtomization: How to begin?

The configuration of the navigation views, card contents and card positions is done in two files named **`views.plist`** and **`cards.plist`** which are xml files adhering to the apple data serialization format (see http://en.wikipedia.org/wiki/Property_list). The basic structure is given by:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
</plist>
```

In Apple's OS X this file format is easily editable in the plist-editor provided with the free Xcode programming environment. As the configuration files are simple UTF-8 encoded files they can be edited with any text editor (e.g. the free Notepad++ for Microsoft Windows).

Now we will walk you through an example customization of iCon.text step by step starting with the example contents embedded into the application.

If you start the application you will be greeted by the embedded example Application.

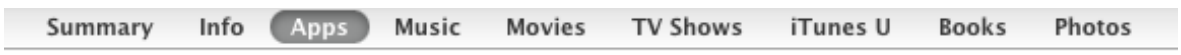
2.1 Installing customized content to your iPad

To customize the application contents and designs the two configuration files and the images for navigation elements and postcards have to be copy into the document folder of the iContext application. This is done via iTunes:

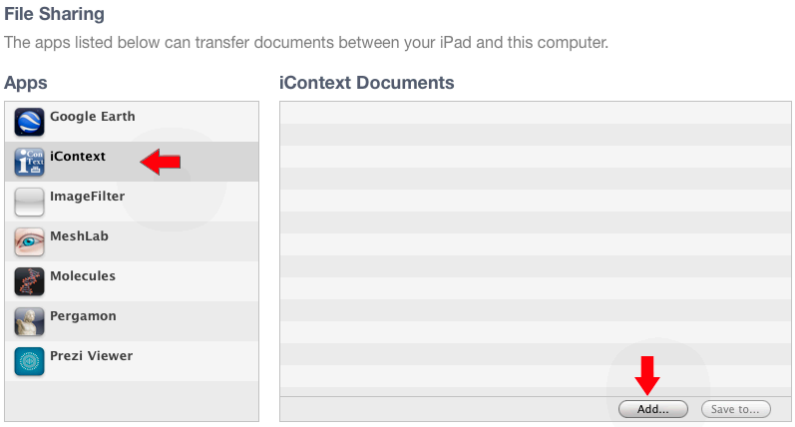
1. Attach the iPad to a computer running iTunes.

2. Select the iPad on left side panel:

3. Navigate to the Apps tab and scroll down to reveal the section File sharing:



4. Select iContext and copy (add) your files into that folder:



5. Press Sync and restart the application on the iPad by double pressing the home button, long tap the iContext icon until a close symbol appears. Tap that icon, leave the multitasking bar by tapping on the home screen or pressing the home button and start iContext again.

2.2 Customizing the navigation bar

The first step is to create a new file called views.plist with

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
     <key>Views</key>
     <array>
     </array>
     <key>Navigation</key>
     <dict>
     </dict>
</dict>
</plist>
```

and copy it to the document folder of the application with itunes.

This results in

Before adding the navigation view buttons you can set the color of the navigation bar to the desired color of your design. Color values are represented as triples consisting of red, green, and blue parts (RGB) with a range of 0.0 to 1.0. White is represented as (1.0,1.0,1.0), black as (0.0,0.0,0.0), while for example pure red is (1.0,0.0,0.0). For the example design we choose a dirty gray value of (0.76,0.75,0.67):

```
…
<key>Navigation</key>
<dict>
     <key>Tint_color</key>
     <array>
          <real>0.76</real>
          <real>0.75</real>
          <real>0.67</real>
          </array>
     </dict>
</dict>
</plist>
…
```

The first button of the navigation bar will be a logo button that provides access to an information screen, that could contain editorial information or credits, for example. Buttons are customized with images in the format png (Portable Network Graphics) which can contain transparent areas. Although the navigation bar height is fixed at 100 pixels, buttons can be larger than that, but are centered so that you have to add a transparent area in the bottom: For a button that should extend 20 pixels into the navigation views the button image has to be 100 pixels+20 pixels (top)+20 pixels

(transparent, bottom)=140 pixels height. The total width of the buttons on the navigation bar should not exceed the 1024 pixel display width.

Example Logo **logo_lang1.png** with transparency marked with checker board pattern and imprint screen imprint.png. The imprint screen will be scaled to fullscreen dimensions of 1024 by 768 pixels.

Logo button.


Imprint screen.

```
…
<key>Views</key>
     <array>
          <dict>
               <key>Type</key>
               <string>Logo</string>
               <key>Button_lang1</key>
               <string>logo-lang1.png</string>
               <key>Imprint_image</key>
               <string>imprint.png</string>
          </dict>
     </array>
…
```

The navigation bar will look after restart of the application like this:

Touching the logo will animate to the imprint screen image:

For a button in another language you can add a second **<key>Button_lang2</key>.**
For more information of creating another language version see section titled ‘Creating another language version’.

2.3 Pile View

To add the button image for the mandatory pile view, a set of two images for an application providing context in a single language or four images for a bilingual kiosk. The two images should inform the user about the active and touchable state of the button.

In this example a transparent button for a single language with a glow for the active state is created:

…

```
            <string>imprint.png</string>
      </dict>
      <dict>
                  <key>Type</key>
                  <string>Pile</string>
                  <key>Background_image</key>
                  <string>background.png</string>
                  <key>Button_lang1_on</key>
                  <string>pile_on_lang1.png</string>
                  <key>Button_lang1_off</key>
                  <string>pile_off_lang1.png</string>
                  <key>Shadow</key>
                  <string>Curl</string>
            </dict>
</array>
…
```

The shadow can be rectangular **`Rect`** or curly **`Curl`**.

To add another language version add **`<key>Button_lang2_on</key>`** and **`<key>Button_lang2</key>`** with the names of the alternative version.

The spacing between the images will be determined by their dimensions.

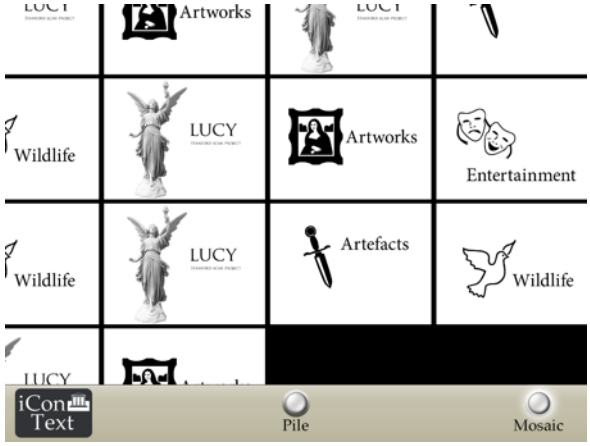
2.4 Mosaic view

Without another navigation view the pile view button it would be not very useful. So let's add another view, the image mosaic view with these two images:


```
…
                <key>Shadow</key>
                <string>Curl</string>
            </dict>
            <dict>
                <key>Type</key>
                <string>Mosaic</string>
                <key>Button_lang1_on</key>
                <string>mosaic_on_lang1.png</string>
                <key>Button_lang1_off</key>
                <string>mosaic_off_lang1.png</string>
            </dict>
</array>
…
```

There are no more configurable parameters in this view.

Now we can switch between these two views. In the mosaic view we can zoom by pinching.



2.5 Categories view

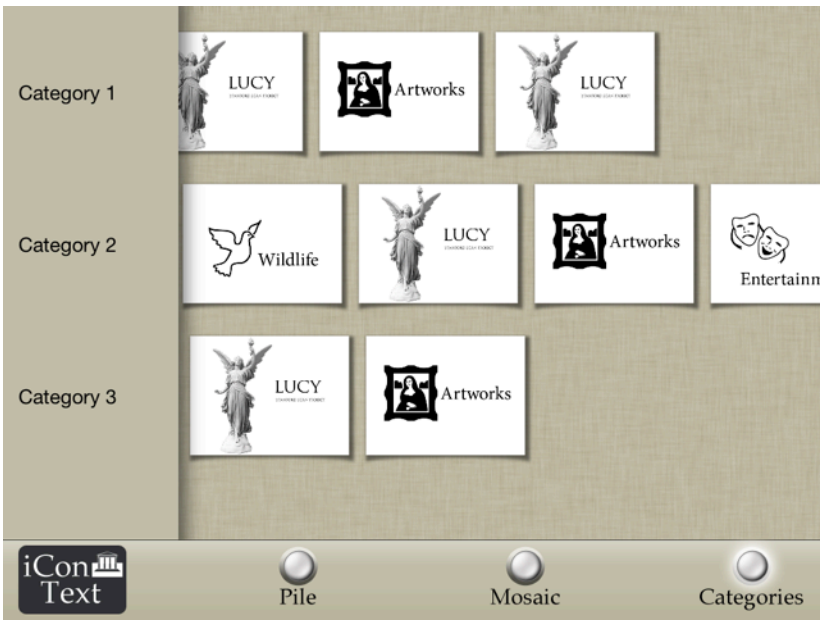
Now we create a navigation view, from which the cards will be grouped by categories. The categories will be called Cat1, Cat2, and Cat. These labels will be used later to categorize the content cards and the entry called l**ang1** will be used for displaying the name. Furthermore we specify a background image and color of text label and text background for the categories.

…

```
            <key>Button_lang1_off</key>
            <string>mosaic_off_lang1.png</string>
      </dict>
      <dict>
            <key>Type</key>
            <string>Categories</string>
            <key>Button_lang1_on</key>
            <string>categories_on_lang1.png</string>
            <key>Button_lang1_off</key>
            <string>categories_off_lang1.png</string>
            <key>Background_image</key>
            <string>backgroundg.png</string>
            <key>Label_background_color</key>
            <array>
                  <real>0.76</real>
                  <real>0.75</real>
                  <real>0.67</real>
            </array>
            <key>Label_text_color</key>
            <array>
                  <real>0.0</real>
                  <real>0.0</real>
                  <real>0.0</real>
            </array>
            <key>Categories</key>
            <array>
                  <dict>
                        <key>Name</key>
                        <string>Cat1</string>
```

```
                        <key>lang1</key>
                        <string>Category 1</string>
                    </dict>
                    <dict>
                        <key>Name</key>
                        <string>Cat2</string>
                        <key>lang1</key>
                        <string>Category 2</string>
                    </dict>
                    <dict>
                        <key>Name</key>
                        <string>Cat3</string>
                        <key>lang1</key>
                        <string>Category 3</string>
                    </dict>
                </array>
            </dict>
</array>
…
```

The number of different categories is not limited, but one card can only belong to one category.

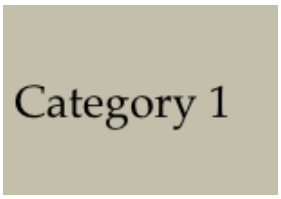


If we would like to switch the order of the navigation buttons we can do so by exchanging the **<dict>…</dict>** blocks of the views.

To change the font to display the category names (and the labels in context view) insert the following into the navigation section to change the font from the default HelveticaNeue to Palatino (for a list of available fonts, search the web for fonts available on iOS devices):

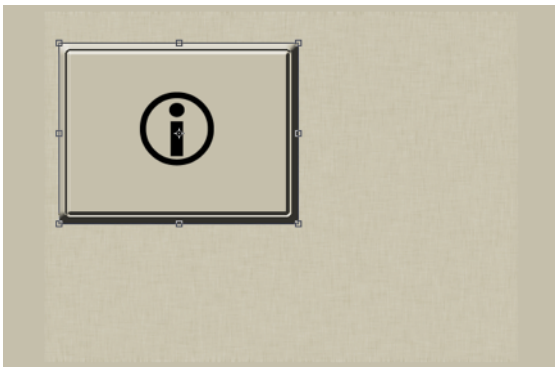
```
…
<key>Navigation</key>
<dict>
```

```
        <key>Tint_color</key>
        <array>
            <real>0.76</real>
            <real>0.75</real>
            <real>0.67</real>
            </array>
            <key>Font</key>
            <string>Palatino</string>
        </dict>
</dict>
</plist>
…
```



2.5 Context view

The context view provides a two-layered access to our postcards: an overview image with touchable areas and optional text labels to access various other images where the postcards are located. Both levels can be pinch zoomed. The postcards can also be hidden by the user to see that particular view undisturbed.
This view could be used for example as access point from a holistic view like a building ensemble, a single building, an era overview, or thematic arrangement of aspects which lead to single architectural buildings, rooms in a building, timelines of objects, or the arrangement of objects belonging to a common theme.
Let us insert a context view with one overview image and one context image: The overview image slightly larger than the iPad screen with dimensions 1200x800 and named `overview.png`. The rectangle should be used to access the context named `Context1`. The rectangle starts at pixel coordinate (116,84) and is 520 pixels wide and 394 pixels high. An optional text label that points at the center of the touchable area should be placed at the pixel coordinate (886,232). The pixel coordinates can be measured in an image editor.



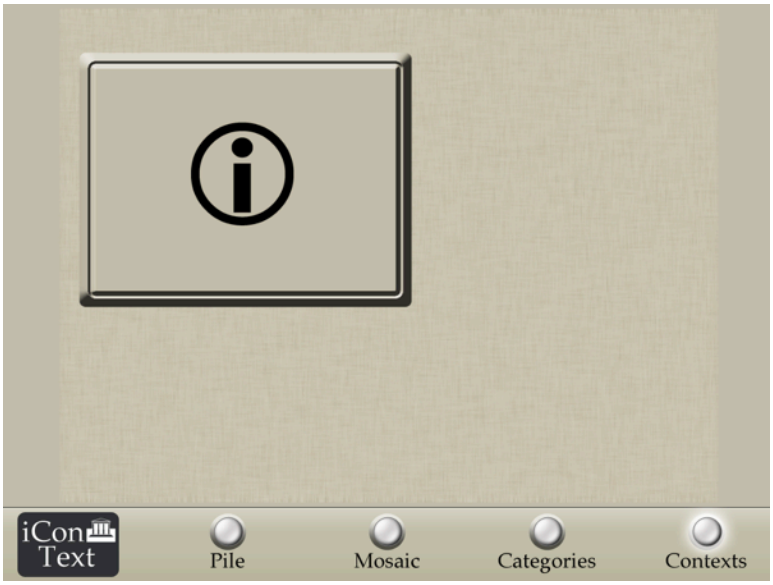
The context image with the same dimensions named `context1.png`:

...

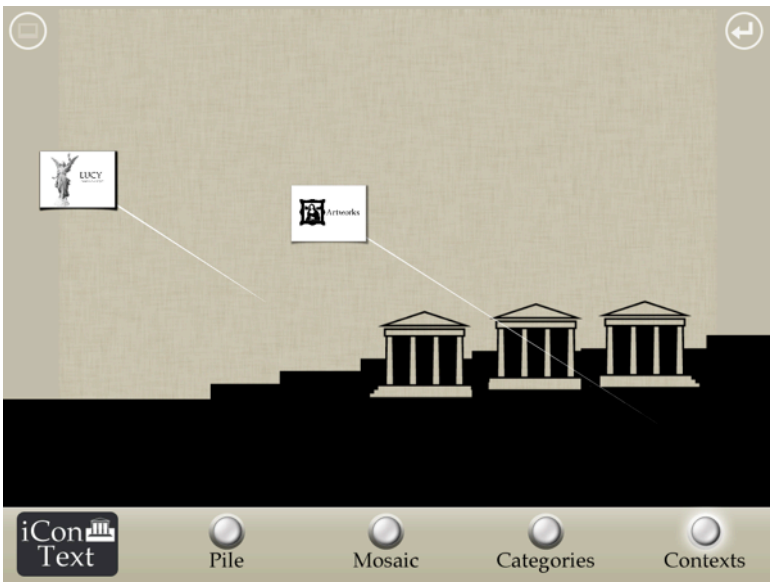
```
                <key>lang1</key>
                        <string>Category 3</string>
                </dict>
        </array>
</dict>
<dict>
        <key>Type</key>
        <string>Context</string>
        <key>Button_lang1_on</key>
        <string>contexts_on_lang1.png</string>
        <key>Button_lang1_off</key>
        <string>contexts_off_lang1.png</string>
        <key>Overview_image</key>
        <string>overview.png</string>
        <key>Card_scale</key>
        <real>0.1</real>
        <key>Contexts</key>
        <array>
                <dict>
                        <key>Context_name</key>
                        <string>Context1</string>
                        <key>Context_image</key>
                        <string>context1.png</string>
                        <key>Show_label</key>
                        <string>NO</string>
                        <key>Label_text_lang1</key>
                        <string>A context label</string>
                        <key>Label_position</key>
                        <array>
                                <integer>886</integer>
                                <integer>232</integer>
                        </array>
                        <key>Touchable_area</key>
                        <array>
                                <integer>116</integer>
                                <integer>84</integer>
                                <integer>520</integer>
                                <integer>394</integer>
                        </array>
                </dict>
        </array>
```

```
            </dict>
</array>
...
```

This results in the navigation view (the label is not shown right now):

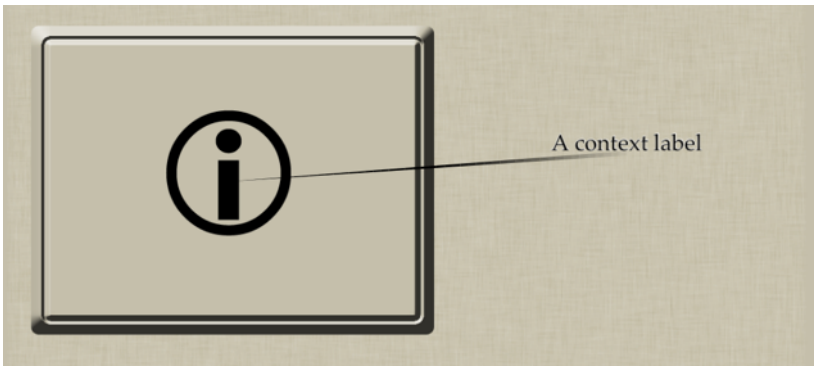


Touching the rectangle accesses the corresponding context view (the position of the postcards will be fixed when customizing the cards):



The button on the top left will hide the postcards, the button on the top right will get the user back to the context overview screen.

If we change the entry of `Show_label` to `YES` the corresponding label will be shown in the the overview image:



Feel free to add as many contexts as you like by adding more <dict>…</dict> sections to the array.

2.6 Plan view

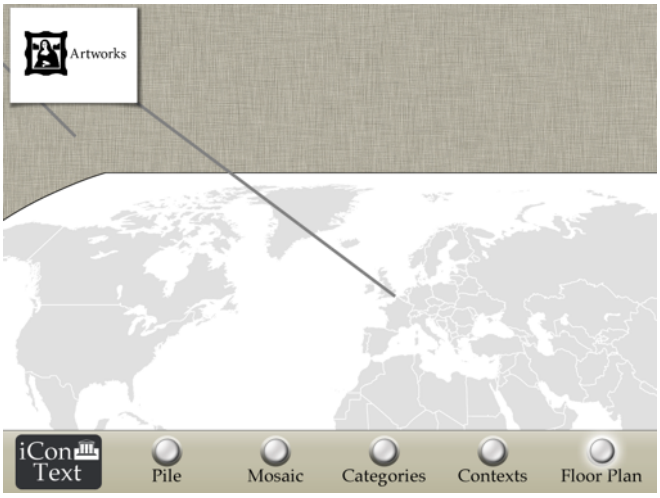
The plan view is essentially providing a single zoomable image from where postcards can be accessed. This could be a floor plan of the exhibition, a map of from where the exhibits come from, or simply a nicely arranged layout for displaying and accessing the artefacts. As floor plan or maps could get really large in pixel dimensions there are two possible ways to embed them into this view. A single image is limited to a resolution of 2048 by 2048 pixels. If you want to show a larger image you should consult the tiled image approach in the appendix.

Let's use an image with dimensions 2048x2048 named map.png and set the background color to gray:

…

```
                <integer>84</integer>
                                <integer>520</integer>
                                <integer>394</integer>
                            </array>
                        </dict>
                </array>
        </dict>
        <dict>
                <key>Type</key>
                <string>Plan</string>
                <key>Button_lang1_on</key>
                <string>plan_on_lang1.png</string>
                <key>Button_lang1_off</key>
                <string>plan_off_lang1.png</string>
```

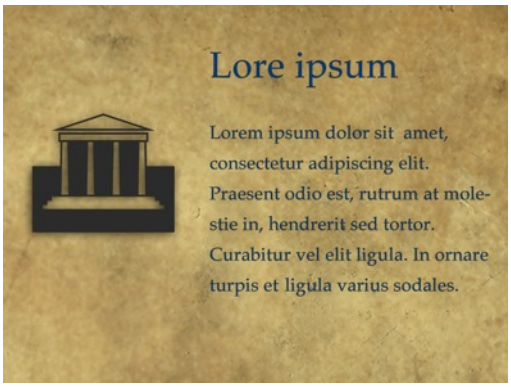
```
                <key>Tiled_image</key>
                <string>NO</string>
                <key>Image_name</key>
                <string>map.png</string>
                <key>Image_dimensions</key>
                <array>
                     <integer>2048</integer>
                     <integer>2048</integer>
                </array>
                <key>Background_color</key>
                <array>
                     <real>0.67</real>
                     <real>0.66</real>
                     <real>0.60</real>
                </array>
           </dict>
</array>
...
```

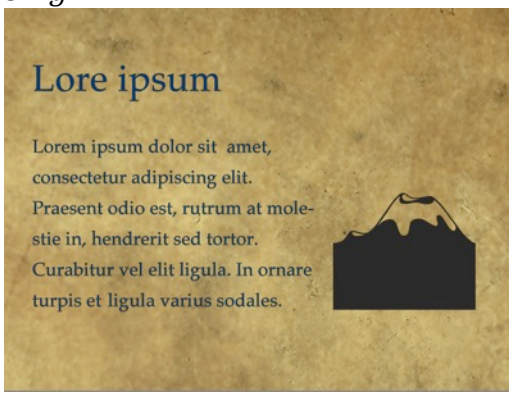


2.7 Creating and customzing cards

Now add one card of our own to the game. We designed an example postcard with front image with pixel dimensions 498 by 372 named `card_front_lang1.png`:

And two backsides with dummy content. The dimension of the back page images has to be *exactly 994 pixels wide and 745 pixels high*.





Create a new file named cards.plist and insert the first card with its content images, category, context, and position therein (the non annotated example will be given in the appendix:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<array>
     <dict>
          <key>name</key>
          <string>Example Card</string>
          This string helps to identify the card for the customizer.
          <key>category</key>
          <string>Cat1</string>
          This string inserts the card into the category is has to have the same name
          as chosen in the views.plist file.
          <key>context</key>
          <string>Context1</string>
          This string inserts the card into the category is has to have
```

the same name as chosen in the views.plist file.

```
<key>card_positions</key>
<dict>
	<key>Context</key>
	<array>
		<integer>350</integer>
		<integer>250</integer>
		<integer>660</integer>
		<integer>568</integer>
	</array>
```

The first pair of number represents the pixel position of the center of the card in the chosen context image. The second pair is the position of the endpoint of the arrow. If you don't want to have an arrow just set it to the same position as the card center.

```
	<key>Plan</key>
	<array>
		<integer>850</integer>
		<integer>300</integer>
		<integer>1050</integer>
		<integer>746</integer>
	</array>
```

The first pair of number are the pixel position of the center of the card in the map image. The second pair is the position of the endpoint of the location line. If you don't want to have a location line just set it to the same position as the card center.

```
</dict>
<key>de</key>
```

The first language version of the card's contents is marked as de for historical reason.

```
<dict>
	<key>frontImage</key>
	<string>card_front_lang1.png</string>
```

The front page image for the front page that is visible in all the views for the first language.

```
	<key>frontImage_hires</key>
	<string>card_front_lang1.png</string>
```

A high resolution version of the front page image that is used in the mosaic view at large zoom scales. If there is no high resolution image available use the same image as above

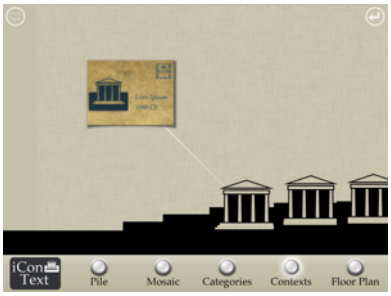
```
	<key>numberOfBackViews</key>
	<integer>2</integer>
```

The number of backside images. There is no limit, but keep in mind that the attention span of the museum visitor is still limited.

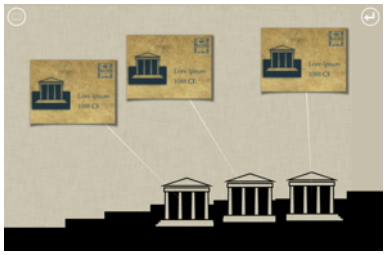
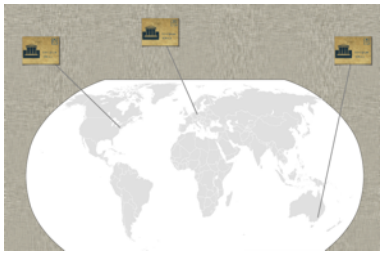
```
	<key>backImages</key>
	<array>
		<string>card_back1_lang1.png</string>
		<string>card_back2_lang1.png</string>

	</array>
	<key>numberOfLinkButtons</key>
```

```
                <integer>0</integer>
```
The number of links that should appear on the back pages. We have none at the moment. Therefore the array of the `linkButtons` should be empty:

```
                <key>linkButtons</key>
                <array>
                </array>
            </dict>
        </dict>
</array>
</plist>
```



By copying this card three times, adjusting the positions in the context and map views and assigning other categories we have a minimal working example configuration:

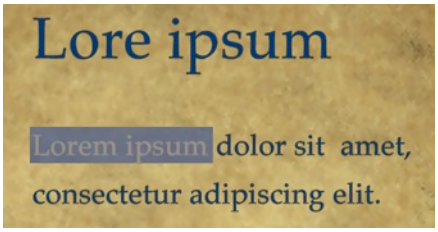



2.8 Adding links

To link from one back page to another postcard you can insert links that will show the linked card. To prevent the museum visitor from getting lost on backsides links on the linked card will not be displayed thus preventing the creation of circle references.

To insert a link for example from the second backside page to the card on position 3 (third in the array) we add the following to the card configuration file :

```
…
<key>numberOfLinkButtons</key>
          <integer>1</integer>
          <key>linkButtons</key>
          <array>
               <array>
               <integer>2</integer>
                 Page number of link.
               <integer>54</integer>
               <integer>246</integer>
                 Upper left corner of link on page image.
               <integer>227</integer>
               <integer>45</integer>
                 Width and height of link area.
               <integer>3</integer>
                 Link to card number #.
               </array>
          </array>
…
```



2.9 Creating another language version

To enable the switching to another language version of your content you can insert a language button that enbles users to toggle between the different version. As an example we localize our app to German. We insert this as the last view in `views.plist`:

```
…
<dict>
                              <key>Type</key>
                              <string>Language</string>
                              <key>Button_lang1</key>
                              <string>lang1.png</string>
                              <key>Button_lang2</key>
                              <string>lang2.png</string>
</dict>
```

…

We then also add another button image for the different navigation buttons. For example the map button (we changed the button text to “Map”, because we are displaying a worldmap as well):

…

```
                        <key>Type</key>
                        <string>Plan</string>
                        <key>Button_lang1_on</key>
                        <string>plan_on_lang1.png</string>
                        <key>Button_lang1_off</key>
                        <string>plan_off_lang1.png</string>
                        <key>Button_lang2_on</key>
                        <string>plan_on_lang2.png</string>
                        <key>Button_lang2_off</key>
                        <string>plan_off_lang2.png</string>
                        <key>Tiled_image</key>
                        <string>NO</string>
```

…

All the other navigation entries are changed accordingly.

For the category view we provide a translation of the category names:

…

```
        <key>Categories</key>
              <array>
                  <dict>
                  <key>Name</key>
                  <string>Cat1</string>
                  <key>lang1</key>
                  <string>Category 1</string>
                  <key>lang2</key>
                  <string>Kategorie 1</string>
                  </dict>
```

…

For the context view we add a translation for each label text:

…

```
        <key>Label_text_lang1</key>
        <string>A context label</string>
        <key>Label_text_lang2</key>
        <string>Kontextbeschriftung</string>
```

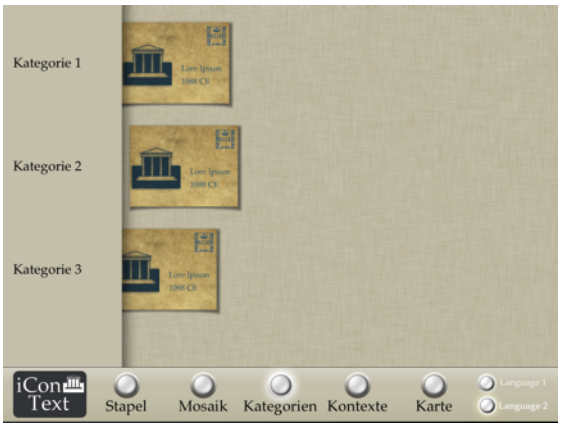
…

Now we have a bilingual navigation bar:

But the postcards appear only as black shadows.

For simplicity we just copy the `<key>de</key><dict>…</dict>` sections and insert them right after these section and change the second key to `<key>**en**</key><dict>…</dict>` without changing the contents in this example. In a real world application we would translate everything and change the images and link positions.

The German version of the category view:



That's about it. We are really curious about new designs and contents and hope that the customization isn't too hard. If you use the application in a museum exhibition we would like to know about your experiences.

If you would like to analyze the usage log files see appendix section 4 "Usage log files" for how to do it.

3. Appendix

3.1 Installing the customization on the device

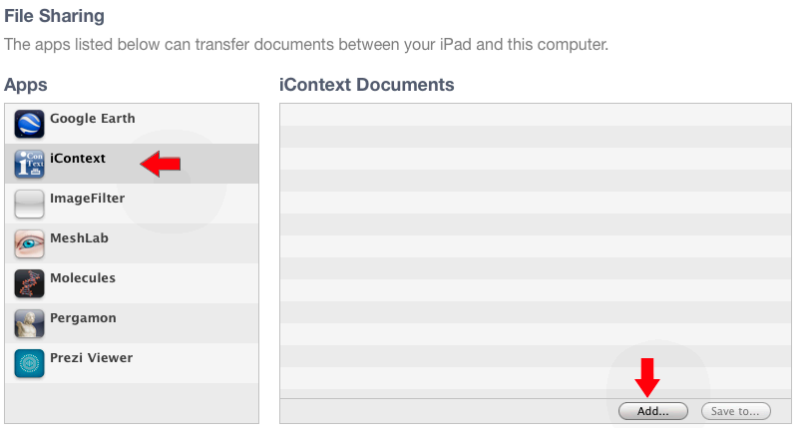
To customize the application contents and designs the two configuration files and the images for navigation elements and postcards have to be copy into the document folder of the iContext application. This is done via iTunes:

1. Attach the iPad to a computer running iTunes.

2. Select the iPad on left side panel:

3. Navigate to the Apps tab and scroll down to reveal the section File sharing:

4. Select iContext and copy (add) your files into that folder:



5. Press Sync and restart the application on the iPad by double pressing the home button, long tap the iContext icon until a close symbol appears. Tap that icon, leave the multitasking bar by tapping on the home screen or pressing the home button and start iContext again.

3.2 Example configuration

`views.plist:`

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>Views</key>
        <array>
                <dict>
                        <key>Type</key>
                        <string>Logo</string>
                        <key>Button_lang1</key>
                        <string>logo_lang1.png</string>
                        <key>Button_lang2</key>
                        <string>logo_lang1.png</string>
                        <key>Imprint_image</key>
                        <string>imprint.png</string>
                </dict>
                <dict>
                        <key>Type</key>
                        <string>Pile</string>
                        <key>Background_image</key>
                        <string>background.png</string>
                        <key>Button_lang1_on</key>
                        <string>pile_on_lang1.png</string>
                        <key>Button_lang1_off</key>
                        <string>pile_off_lang1.png</string>
                        <key>Button_lang2_on</key>
                        <string>pile_on_lang2.png</string>
                        <key>Button_lang2_off</key>
                        <string>pile_off_lang2.png</string>
                        <key>Shadow</key>
                        <string>Curl</string>
                 </dict>
                 <dict>
                        <key>Type</key>
                        <string>Mosaic</string>
                        <key>Button_lang1_on</key>
                        <string>mosaic_on_lang1.png</string>
```

```
        <key>Button_lang1_off</key>
        <string>mosaic_off_lang1.png</string>
        <key>Button_lang2_on</key>
        <string>mosaic_on_lang2.png</string>
        <key>Button_lang2_off</key>
        <string>mosaic_off_lang2.png</string>
</dict>
<dict>
         <key>Type</key>
         <string>Categories</string>
         <key>Button_lang1_on</key>
         <string>categories_on_lang1.png</string>
         <key>Button_lang1_off</key>
         <string>categories_off_lang1.png</string>
         <key>Button_lang2_on</key>
         <string>categories_on_lang2.png</string>
         <key>Button_lang2_off</key>
         <string>categories_off_lang2.png</string>
         <key>Background_image</key>
         <string>background.png</string>
         <key>Label_background_color</key>
         <array>
         <real>0.76</real>
         <real>0.75</real>
         <real>0.67</real>
         </array>
         <key>Label_text_color</key>
         <array>
         <real>0.0</real>
         <real>0.0</real>
         <real>0.0</real>
         </array>
         <key>Categories</key>
         <array>
                <dict>
                <key>Name</key>
                <string>Cat1</string>
                <key>lang1</key>
                <string>Category 1</string>
                <key>lang2</key>
                <string>Kategorie 1</string>
                </dict>
                <dict>
                <key>Name</key>
                <string>Cat2</string>
                <key>lang1</key>
                <string>Category 2</string>
                <key>lang2</key>
                <string>Kategorie 2</string>
                </dict>
                <dict>
                <key>Name</key>
                <string>Cat3</string>
                <key>lang1</key>
                <string>Category 3</string>
                <key>lang2</key>
                <string>Kategorie 3</string>
                </dict>
         </array>
 </dict>
 <dict>
        <key>Type</key>
        <string>Context</string>
        <key>Button_lang1_on</key>
        <string>contexts_on_lang1.png</string>
        <key>Button_lang1_off</key>
        <string>contexts_off_lang1.png</string>
        <key>Button_lang2_on</key>
        <string>contexts_on_lang2.png</string>
        <key>Button_lang2_off</key>
        <string>contexts_off_lang2.png</string>
```

```
                <key>Overview_image</key>
                <string>overview.png</string>
                <key>Card_scale</key>
                <real>0.2</real>
                <key>Contexts</key>
                <array>
                        <dict>
                        <key>Context_name</key>
                        <string>Context1</string>
                        <key>Context_image</key>
                        <string>context1.png</string>
                        <key>Show_label</key>
                        <string>NO</string>
                        <key>Label_text_lang1</key>
                        <string>A context label</string>
                        <key>Label_text_lang2</key>
                        <string>Kontextbeschriftung</string>
                        <key>Label_position</key>
                        <array>
                                <integer>886</integer>
                                <integer>232</integer>
                        </array>
                        <key>Touchable_area</key>
                        <array>
                                <integer>116</integer>
                                <integer>84</integer>
                                <integer>520</integer>
                                <integer>394</integer>
                        </array>
                        </dict>
                </array>
        </dict>
        <dict>
                <key>Type</key>
                <string>Plan</string>
                <key>Button_lang1_on</key>
                <string>plan_on_lang1.png</string>
                <key>Button_lang1_off</key>
                <string>plan_off_lang1.png</string>
                <key>Button_lang2_on</key>
                <string>plan_on_lang2.png</string>
                <key>Button_lang2_off</key>
                <string>plan_off_lang2.png</string>
                <key>Tiled_image</key>
                <string>NO</string>
                <key>Image_name</key>
                <string>map.png</string>
                <key>Image_dimensions</key>
                <array>
                        <integer>2048</integer>
                        <integer>2048</integer>
                </array>
                <key>Background_color</key>
                <array>
                        <real>0.67</real>
                        <real>0.66</real>
                        <real>0.60</real>
                </array>
                </dict>
        <dict>
                <key>Type</key>
                <string>Language</string>
                <key>Button_lang1</key>
                <string>lang1.png</string>
                <key>Button_lang2</key>
                <string>lang2.png</string>
        </dict>
</array>
<key>Navigation</key>
<dict>
        <key>Tint_color</key>
```

```
                <array>
                        <real>0.76</real>
                        <real>0.75</real>
                        <real>0.67</real>
                </array>
                <key>Font</key>
                <string>Palatino</string>
        </dict>
</dict>
</plist>
```

cards.plist:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<array>
       <dict>
              <key>name</key>
              <string>Example Card1</string>
              <key>category</key>
              <string>Cat1</string>
              <key>context</key>
              <string>Context1</string>
              <key>card_positions</key>
              <dict>
                     <key>Context</key>
                     <array>
                            <integer>350</integer>
                            <integer>250</integer>
                            <integer>660</integer>
                            <integer>568</integer>
                     </array>
                     <key>Plan</key>
                     <array>
                            <integer>850</integer>
                            <integer>300</integer>
                            <integer>1050</integer>
                            <integer>746</integer>
                     </array>
              </dict>
              <key>de</key>
              <dict>
                     <key>frontImage</key>
                     <string>card_front_lang1.png</string>
                     <key>frontImage_hires</key>
                     <string>card_front_lang1.png</string>
                     <key>numberOfBackViews</key>
                     <integer>2</integer>
                     <key>backImages</key>
                     <array>
                            <string>card_back1_lang1.png</string>
                            <string>card_back2_lang1.png</string>
                     </array>
                     <key>numberOfLinkButtons</key>
                     <integer>1</integer>
                     <key>linkButtons</key>
                     <array>
                         <array>
                         <integer>2</integer>
                         <integer>54</integer>
                         <integer>246</integer>
                         <integer>227</integer>
                         <integer>45</integer>
                         <integer>2</integer>
                         </array>
                     </array>
              </dict>
              <key>en</key>
              <dict>
```

```
                <key>frontImage</key>
                <string>card_front_lang1.png</string>
                <key>frontImage_hires</key>
                <string>card_front_lang1.png</string>
                <key>numberOfBackViews</key>
                <integer>2</integer>
                <key>backImages</key>
                <array>
                        <string>card_back1_lang1.png</string>
                        <string>card_back2_lang1.png</string>
                </array>
                <key>numberOfLinkButtons</key>
                <integer>1</integer>
                <key>linkButtons</key>
                <array>
                    <array>
                    <integer>2</integer>
                    <integer>54</integer>
                    <integer>246</integer>
                    <integer>227</integer>
                    <integer>45</integer>
                    <integer>2</integer>
                    </array>
                </array>
        </dict>
</dict>
<dict>
        <key>name</key>
        <string>Example Card2</string>
        <key>category</key>
        <string>Cat2</string>
        <key>context</key>
        <string>Context1</string>
        <key>card_positions</key>
        <dict>
                <key>Context</key>
                <array>
                        <integer>615</integer>
                        <integer>180</integer>
                        <integer>838</integer>
                        <integer>555</integer>
                </array>
                <key>Plan</key>
                <array>
                        <integer>200</integer>
                        <integer>400</integer>
                        <integer>630</integer>
                        <integer>818</integer>
                </array>
        </dict>
        <key>de</key>
        <dict>
                <key>frontImage</key>
                <string>card_front_lang1.png</string>
                <key>frontImage_hires</key>
                <string>card_front_lang1.png</string>
                <key>numberOfBackViews</key>
                <integer>2</integer>
                <key>backImages</key>
                <array>
                        <string>card_back1_lang1.png</string>
                        <string>card_back2_lang1.png</string>
                </array>
                <key>numberOfLinkButtons</key>
                <integer>0</integer>
                <key>linkButtons</key>
                <array>
                </array>
        </dict>
        <key>en</key>
        <dict>
```

```
                <key>frontImage</key>
                <string>card_front_lang1.png</string>
                <key>frontImage_hires</key>
                <string>card_front_lang1.png</string>
                <key>numberOfBackViews</key>
                <integer>2</integer>
                <key>backImages</key>
                <array>
                        <string>card_back1_lang1.png</string>
                        <string>card_back2_lang1.png</string>
                </array>
                <key>numberOfLinkButtons</key>
                <integer>0</integer>
                <key>linkButtons</key>
                <array>
                </array>
        </dict>
</dict>
<dict>
        <key>name</key>
        <string>Example Card3</string>
        <key>category</key>
        <string>Cat3</string>
        <key>context</key>
        <string>Context1</string>
        <key>card_positions</key>
        <dict>
                <key>Context</key>
                <array>
                        <integer>985</integer>
                        <integer>160</integer>
                        <integer>1005</integer>
                        <integer>545</integer>
                </array>
                <key>Plan</key>
                <array>
                        <integer>1900</integer>
                        <integer>400</integer>
                        <integer>1706</integer>
                        <integer>1306</integer>
                </array>
        </dict>
        <key>de</key>
        <dict>
                <key>frontImage</key>
                <string>card_front_lang1.png</string>
                <key>frontImage_hires</key>
                <string>card_front_lang1.png</string>
                <key>numberOfBackViews</key>
                <integer>2</integer>
                <key>backImages</key>
                <array>
                        <string>card_back1_lang1.png</string>
                        <string>card_back2_lang1.png</string>
                </array>
                <key>numberOfLinkButtons</key>
                <integer>0</integer>
                <key>linkButtons</key>
                <array>
                </array>
        </dict>
        <key>en</key>
        <dict>
                <key>frontImage</key>
                <string>card_front_lang1.png</string>
                <key>frontImage_hires</key>
                <string>card_front_lang1.png</string>
                <key>numberOfBackViews</key>
                <integer>2</integer>
                <key>backImages</key>
                <array>
```

```
                        <string>card_back1_lang1.png</string>
                        <string>card_back2_lang1.png</string>
                    </array>
                    <key>numberOfLinkButtons</key>
                    <integer>0</integer>
                    <key>linkButtons</key>
                    <array>
                    </array>
                </dict>
        </dict>
</array>
</plist>
```

3.3 Using a tiled image in the plan view

Not yet documented.

3.4 Using the usage log files

After being used for some time you may have a look at the text files created by the application in the file sharing folder with itunes. These log files save the timestamps of session start and resets. From these you can count the number of and duration of sessions in which the application has been used.

Example:

`20120703:11:29:15.86:iPad:SESSION RESET` **start** of application
`20120703:11:31:17.80:iPad:SESSION START` **start of first session**
`20120703:11:36:22.32:iPad:SESSION RESET` **end** of session, duration ~ 5 minutes
`20120703:11:46:07.12:iPad:SESSION START` **start** of second session
`20120703:11:49:07.95:iPad:SESSION RESET` **end** of session, duration ~3 minutes

3.5 Troubleshooting

Empty design/content:

Make sure the configuration files are correct and complete. See example.

Navigation button not visible:

Check if images are there and referenced correctly in the configuration files.

Cards appear as gray shadows:

Make sure the card images are there and referenced correctly.

Back side images look blurry/distorted:

Make sure the images have the correct pixel dimension of 994x745.

App crashes when trying to access X:

Make sure the syntax of the configuration files is followed correctly. You may have forgotten some entry even though it's not used and has to be set to a sane value.

3.6 Acknowledgements

iCon.text was developed for the presentation of results from the “Berlin Sculpture Network” at the Pergamon special exhibition at the Pergamon Museum Berlin 2011/2012 and funded by the German Ministry of Education and Research (BMBF) with grant agreement U809068.