

Technical Notes - AcehCamera v1.00 - 07/11/2006

Known errors/issues and additions needed:

- Program will freeze or crash when sending/receiving if there is no network connection or network connection is lost. Reason: there is no error prevention in place to counter this yet.
- Downloading data for "play" takes a lengthy amount of time. This is because of the way the alpha data is stored (as a string) and the phone must split the received data into an array. Hence processing takes awhile.
- Data is not always saved successfully to the database (ie not saved at all, or in rare cases empty values are placed in the database record even though it checks for null). Specifically when new image creations are made or edited. Reasoning for this error may be related to this: <http://forum.java.sun.com/thread.jspa?threadID=778390>
- There is no transaction-processing algorithm in place to prevent 2 or more users from simultaneously editing the same image creation. Cannot be implemented until the data is always successfully sent, or image creation instances may be permanently blocked to users because the database thinks they are still in use.
- Camera sometimes throws "Uncaught Exceptions", on phone only (never happened in emulator mode). This only seems to happen when the camera is used for a second/third time by the application without the application being closed. The camera is properly disposed of after use by java each time, so I can only suggest it is an issue with the Nokia implementation of the MMAPI.
- Source cannot be run in emulator. Needs a camera to produce images of N80 size, also runs out of memory with default settings (due to amount of data being downloaded and size of images being displayed).
- Error checking is not put in place for when there is less than 2 images in the database or 0 image creations.
- Menu options need to be written in Acehnese. A translator or dictionary is required.

Solutions and explanation of alpha channel problem:

Currently the alpha data is passed between phone and server in a String. When the phone starts a new "Creation" it creates an array of integers for every pixel (ie 320*240 pixels =76800 values) for both the first and second image layers. This is quite processor and memory intensive but it allows the phone to easily locate pixels that need their alpha values adjusted, as specified by the user. The phone updates the images onscreen by using the getRGB() method on the Image objects to get an array of ARGB pixel data. The array of integers is then combined with the pixel data by bit-shifting the corresponding

alpha value into place. The Image object is then visually recreated with the new array of pixels.

When the phone sends data to the server in order to save the Creation, each array of integers is cast to a String delimited by commas. Both Strings are sent to the server, which stores them as text files with a unique file name. They are stored as text files because the MySQL database will not store so many characters in a varchar. These Strings are sent back to the phone when the application is in "Play" mode. Not only does the phone have to process receiving 3 jpg images (these are kept at a small size) and two large Strings, it must manually split the String file data into an array of integers so it can be used for alpha channel processing.

This current implementation is memory and processor intensive. The String text files that are saved are roughly 300kb, which is very large! In searching for a more elegant solution to this problem, which would also hopefully solve interrupted data transmissions, I concluded that the other way to pass alpha data between phone and server is within an image file, namely PNG. PNG files are supported by phones and are good at displaying photos as well as supporting transparency (alpha channels). In initial tests of the project I found that using the server and PHP to resize camera images and save them as PNGs produced files of around 100kb. This was deemed far too large and so I switched to jpg images, which were about 20kb at most. However, in relation to the alpha channel problem 100kb is far better than 300kb.

So I suggested to myself that the phone could use arrays of integers to help process images as they were edited by the user, but then send back the Image objects as PNG files instead of using Strings. These PNG files could be stored on the server and returned to the phone when called for. Because PNG can save transparency data the phone would be able to extract the alpha channel data via bit-shifting into an array of Integers and carry on as if it had just downloaded the same data in a String.

The problem with this solution lies in the fact there is no easy way to encode a PNG in J2ME. The Image object will not let you extract the byte[] data with PNG encoding, the only time you can get this from J2ME is when you use the camera to take a snapshot. A search of the internet returns no easy solution to this problem. This website details the difficulties in encoding PNG files with J2ME:
<http://home.planet.nl/~beems135/bv/png.html>

Of course it is an option to send back just the pixel data and let PHP save the image as PNG, but this won't account for any savings in the mobile's sending of alpha data, in fact in this scenario it would be sending more data as not just alpha channel information but pixel information would be sent too.

In conclusion, the more elegant design is feasible (as it is not impossible) but it is not within my capabilities to implement given the time frame for working on the prototype. A greater knowledge of the algorithms behind image formats, as well as the capabilities

of J2ME and the processing cost of using compression/encoding techniques will be needed to tackle the problem.

Follow on problem - The web gallery:

One of the other goals of my project is to provide a web interface that allows people from around the world to view what the children of Aceh create. Each Creation will have a limited lifespan as there are only so many pixel chunks you can remove. A cycle of this lifespan will entail the usage by one child to edit the creation (they are limited in the number of changes they can make) and then save the results to the server. The server is supposed to also save a compiled image at this point that represents what the child saw when they saved the Creation. The compiled images for each Creation are kept together, related by file name, and ordered ascending by date (it would be most efficient if they had their own database table). These compiled images are permanent and are to be used to show the world the Creations of the children and how they have evolved. For example, if an image goes through 20 cycles in its life we will be able to show what it looked like after each cycle, which will show how it developed and changed.

The compiled images are designed to be displayed sequentially or in a slow animated fashion. The web site might make use of the average colour information gathered to group or navigate through these Creation sets.

The problem and reason this part of the project has not yet been implemented is there is no easy way to output picture of a Creation at one point in its life cycle. PHP has the ability to blend alpha channels, so it is possible to create an image from multiple images that contain alpha channels. This would be easiest if the phone could output PNG files, the reason this can't be implemented is discussed in the above section. The other way considering the available data is to read in the alpha channel String files, split each one using the delimiter into an array and then apply the alpha data to the pixel data. PHP does not offer a method to easily grab all pixel data; this must be done manually pixel by pixel. There is also no way to create an image based on an array of pixels. You can manually loop through all pixels and allocate the colour (this only accepts red, green and blue values separately - no alpha) or you can use the set pixel method, which accepts 'colour' as int.

This int may or may not allow you to include alpha data, I haven't been able to successfully finish researching how to use PHP to accomplish saving these 'end of cycle Creation images'. If all else fails it should be possible to individually set each pixel in a new Image instance based on what the alpha data says the pixel should be by manually checking. I was hoping to be able to research a way for J2ME to save PNG images, as using PHP to save an image based on three alpha blended PNG images is much simpler than jumping through hoops to calculate transparency values individually.

The result, because my implementation of J2ME PNG saving cannot go ahead currently, I have been unable to set up PHP to save a Creation at each stage in its cycle and hence cannot supply a website which displays these images to the general public in this

implementation of the prototype. This part of the implementation will not directly affect the users of the mobile application, but won't allow the fulfilment of their need to spread their stories to the outside world. It is certainly feasible for this aspect to be implemented in future prototypes/final product.

Implementation of Camera for average colour:

My original idea for the average colour was to implement the camera so data could be read and the average colour displayed on the fly onscreen. This would reduce confusion to the user at having two menu options that used the camera interface. However it is not possible to grab data from the camera without doing a snapshot, meaning on the fly updating is impossible. Another way to implement the interface to reduce initial confusion would be to have one menu option and after the user takes a picture they can choose (from thumbnails onscreen) to keep the photo or send the average colour to the server.

- Amanda Bell
200321378

mySQL Database information:

```
+-----+
| Tables_in_abel2906 |
+-----+
| Creations          |
| Images             |
+-----+
```

Creations

Field	Type	Null	Key	Default	Extra
id	int(11)		PRI	NULL	auto_increment
image1	int(11)	YES		NULL	
image2	int(11)	YES		NULL	
image3	int(11)	YES		NULL	
alpha1	varchar(100)	YES		NULL	
alpha2	varchar(100)	YES		NULL	
playCount	int(11)	YES		NULL	

Images

Field	Type	Null	Key	Default	Extra
imageID	int(11)		PRI	NULL	auto_increment
name	varchar(100)	YES		NULL	
avgRed	int(11)	YES		NULL	
avgGreen	int(11)	YES		NULL	
avgBlue	int(11)	YES		NULL	

- Foreign key from Creations.image1, Creations.image2, Creations.image3 to Image.imageID.
- Creations.alpha1, Creations.alpha2 and Images.name are unique file identifiers.