香港中文大學
計算機科學及工程學系

Department of Computer Science and Engineering,
The Chinese University of Hong Kong

IK0901 – iPhone Development for Education and Social Networking

Group Members

Tang Chi Chiu, James 08601684

Supervised By

Prof. Irwin King

ABSTRACT

Recalling the project goal that was defined in semester 1, the main objective is to create a mind map application on iPhone utilized the unique features of it in order to enhance the capability of mind map on iPhone.

This application is aimed at helping students in learning during or after lectures as the initiative motivation. As the research in semester 1 indicated that it is lack of good utility program on iPhone in improving the efficiency of learning for students. It means there is a greater market potential in doing such an application or project.

The idea of implement Mind map application was chosen as the goal. It is because mind map in nature is a tool in helping people to organize thought. Researches in semester 1 also indicated there were only a handful mind map applications on iPhone. Most of them didn't utilize the feature of iPhone. As time proceeds, especially after iPhone OS 3.0 came out, more and more new features of iPhone can be applied to mind map application, yet there were no updates of existing iPhone mind application try to employ them that is planned to employ in this project.

At the time being of the submission of this report, the implementation of the application features are done completely, the application is ready to submit to Apple's App Store and should be in the way of approval.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

In the very first parts of the report, there is an introduction of the project to describe our motivation and objective in more detail.

In the second part, there is a brief introduction of what is mind map as well as the usage and background of it. Following some researches that are done through out these two semesters. This is mainly a recap section, therefore, a recap of the work that were done in semester 1 would be mention to help the reader get a better understanding of the whole progress of this project.

The following sections are mainly the design and implementation details of the application. The content in these sections would be varied in each individual report as required. Each individual will explain much detail in his or her design as well as implementation. A chapter that is dedicates to explain the division of labor is placed at the beginning before the design and implementation details.

After the design and implementation sections, there is a quick experiments evaluation of the application. It is focus on the performance of the final build of the application.

Finally, there is a conclusion to wrap up to show what have been achieved in this project. A brief (not in very technical details) section to describe the difficulties and future work/planning is included in the conclusion as well.

1. INTRODUCTION

The project goal is to create a mind map application on iPhone that cooperate with the unique features of iPhone to enhance the capability of the usage of mind map. This application is aimed to help students in learning during or after lectures as the initiative motivation. Different techniques were introduced by learners to help different people in memorizing and revising information. As the vast usage of mind map not just limited to learning but helping us in organizing stuffs in many ways. A mind map application combines with iPhone features is the project goal.

MOTIVATION

As the project topic is about "iPhone Development for Education and Social Networking". Many ideas that could benefit users were come up immediately, Eventually, developing an Application that is just easy for user to draw Mindmaps became the project goal. There are many Mindmap applications on esktop environment as well as on the iPhone App Store, but none of them can really replace the traditional way of drawing mindmaps on a piece of paper. All of them are limited in some ways in drawing flexibilities, functionalities, and most importantly, only a limited capabilities of this magical iPhone platform are being utilized within those Applications.

BACKGROUND

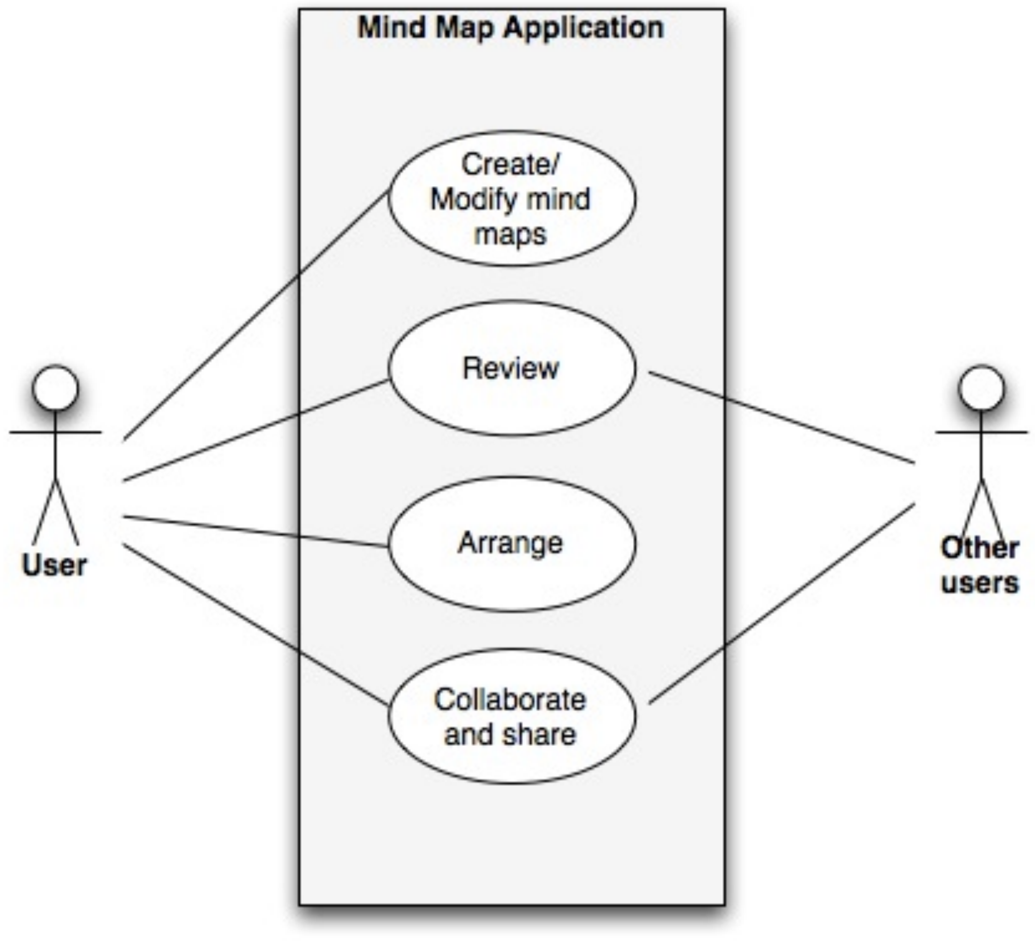
So, what is mind map?

A mind map is a diagram used to represent words, ideas, tasks, or other items linked to and arranged around a central key word or idea. Mind maps are used to generate, visualize, structure, and classify ideas, and as an aid in study, organization, problem solving, decision making, and writing. The visual basis of them helps one to distinguish words or ideas, often with colors and symbols. They generally take a hierarchical or tree branch format, with ideas branching into their subsections. Mind maps allow for greater creativity when recording ideas and information, as well as allowing the note-taker to associate words with visual representations.

TARGET USERS

Focus on students in the development. Generally, most of the advanced features are customized for the basic uses of mind map e.g. organize ideas, taking notes during lectures.

The following is a Use-case Diagram that introduces the basic function of the application and what the application can be provided to the user.



MIND MAP USAGE

A mind map is often created around a single word or text, placed in the center, to which associated ideas, words and concepts are added. Mind maps can be drawn by hand, either as 'rough notes' during a lecture or meeting, for example, or can be more sophisticated in quality. Examples of both are illustrated. There are also a number of software packages available for producing mind maps.

Mind maps have many applications in personal, family, educational, and business situations, including note taking, brainstorming (wherein ideas are inserted into the map radically around the center node, without the implicit prioritization that comes from hierarchy or sequential arrangements, and wherein grouping and organizing is reserved for later stages), summarizing, revising, and general clarifying of thoughts. One could listen to a lecture, for example, and take down notes using mind maps for the most important points or keywords. One can also use mind maps as a mnemonic technique or to sort out a complicated idea. Mind maps are also promoted as a way to collaborate in color pen creativity sessions.

Mind maps can be used for:

- Problem Solving
- Outline / Framework Design
- Anonymous collaboration.
- Marriage of words and visuals.
- Individual expression of creativity.
- Condensing material into a concise and memorable format.
- Team building or synergy creating activity.
- Enhancing work morale.

COMPARISON

Since last semester, we couldn't find more free mind map applications

Features	**Mindblowing**	**SimpleMindX**	**ThinkingMap Lite**
Overview and managing mind maps			
Thumbnail preview	X	X	
Custom arrangement		X	
Mind map management	X	X	
Mind map Details			
Colorized nodes	X	X	
Moving nodes	X	X	
Nodes auto arrangement	X		X
Connecting nodes	X	X	X
Apply visual styles		X	
Audio nodes	X		X
Video nodes	X		X
Image nodes	X		X
Zoom	X	X	
Scroll	X	X	
Rotate	X	X	X
Nodes descriptions	X		

List view of nodes in tree structure	X		

Other Options

Export to SimpleMind		X	
Export to Bundled Application		X	X
Cut/Copy	X	X	
Paste	X	X	
Export to email	X		X
Export to Camera roll	X	X	
Undo/Redo	X	X	

RECAP

Completed functionalities compare with semester 1.

Features	Semester 1	Semester 2
Overview and managing mind maps		
Mind map management	X	X
Thumbnail preview		X
Mind map Details		
Colorized nodes	X	X
Moving nodes	X	X
Nodes auto arrangement		X
Connecting nodes	X	X
Audio nodes		X
Video nodes		X
Image nodes		X
Zoom	X	X
Scroll	X	X
Rotate		X
Nodes descriptions	X	
List view of nodes	X	
Other Options		

Cut/Copy		X
Paste		X
Export to email		X
Export to Camera roll		X
Undo/Redo		X

Division of labor

The division is mostly divided base on the feature. However, there are also a few back-end and GUI implementation within.

Features	Tsang Ho Kwan 08601694	Tang Chi Chiu 08601684
Overview and managing mind maps		
Mind map management		X
Thumbnail preview	X	X
Mind map Details		
Colorized nodes	X	
Moving nodes	X	
Nodes auto arrangement	X	
Connecting nodes	X	
Audio nodes		X
Video nodes		X
Image nodes		X
Zoom	X	
Scroll	X	
Rotate	X	X
Nodes descriptions	X	X
List view of nodes		X
Other Options		

Cut/Copy	X	
Paste	X	
Export to email	X	X
Export to Camera roll		X
Undo/Redo	X	

Back-end and miscellaneous

Database		X
Mind map drawing mechanism	X	
Overall GUI design	X	X

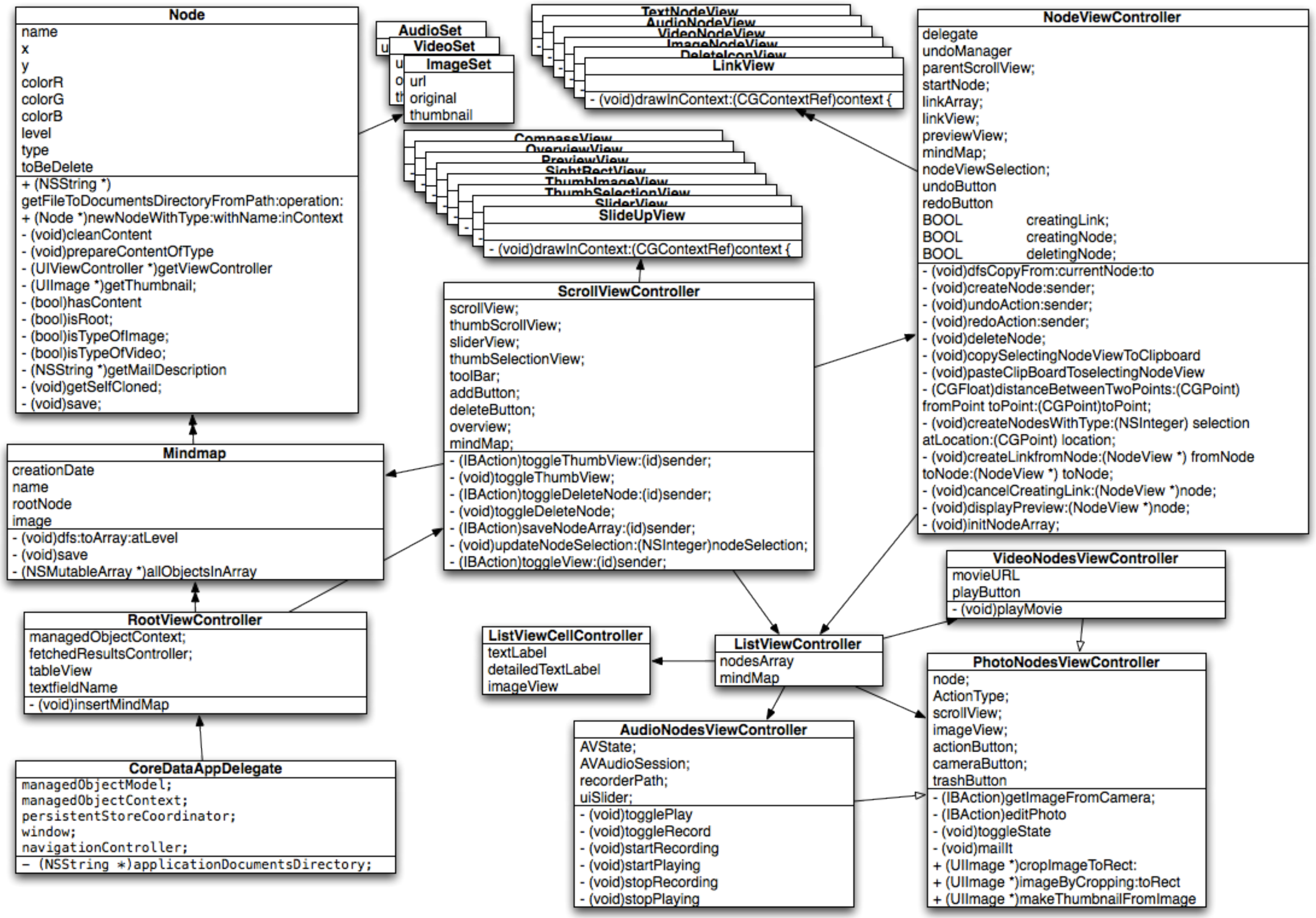
2. OVERALL DESIGN

SYSTEM ARCHITECTURE

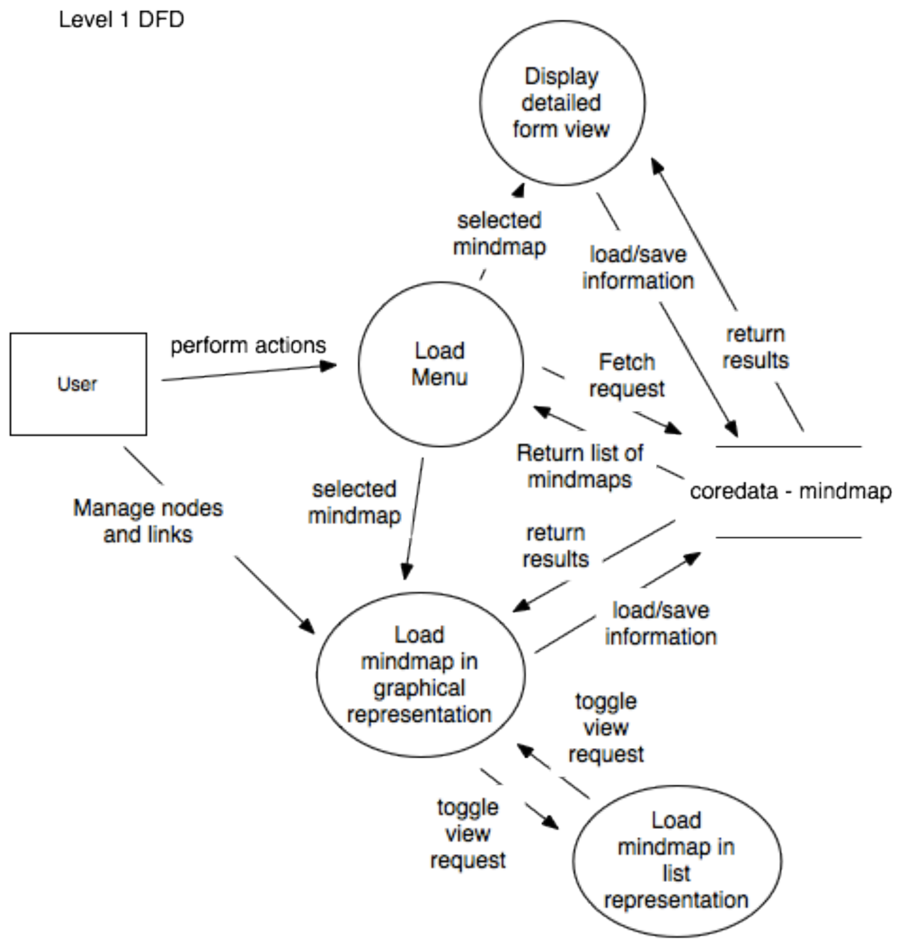
We will be using UML to represent our system architecture. And we have the following graphs.

- The Class diagram
- The DFD diagram
- The Program flow

CLASS DIAGRAM

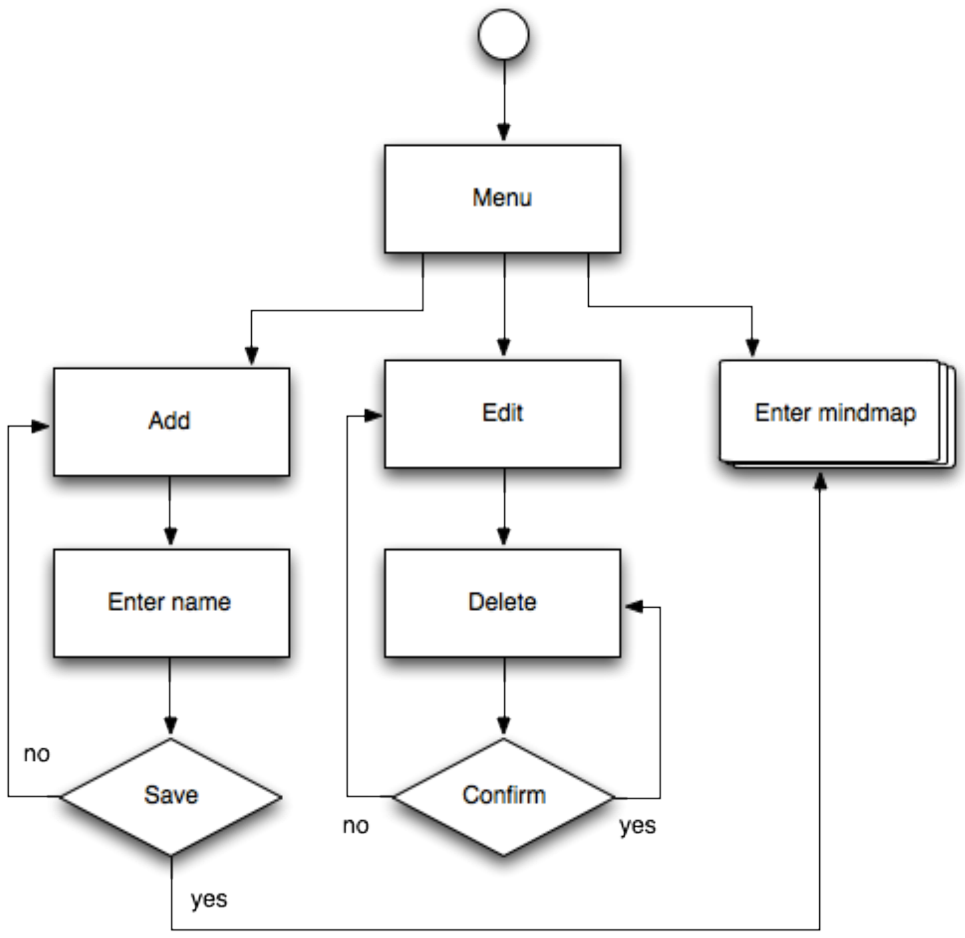


The DFD Diagram



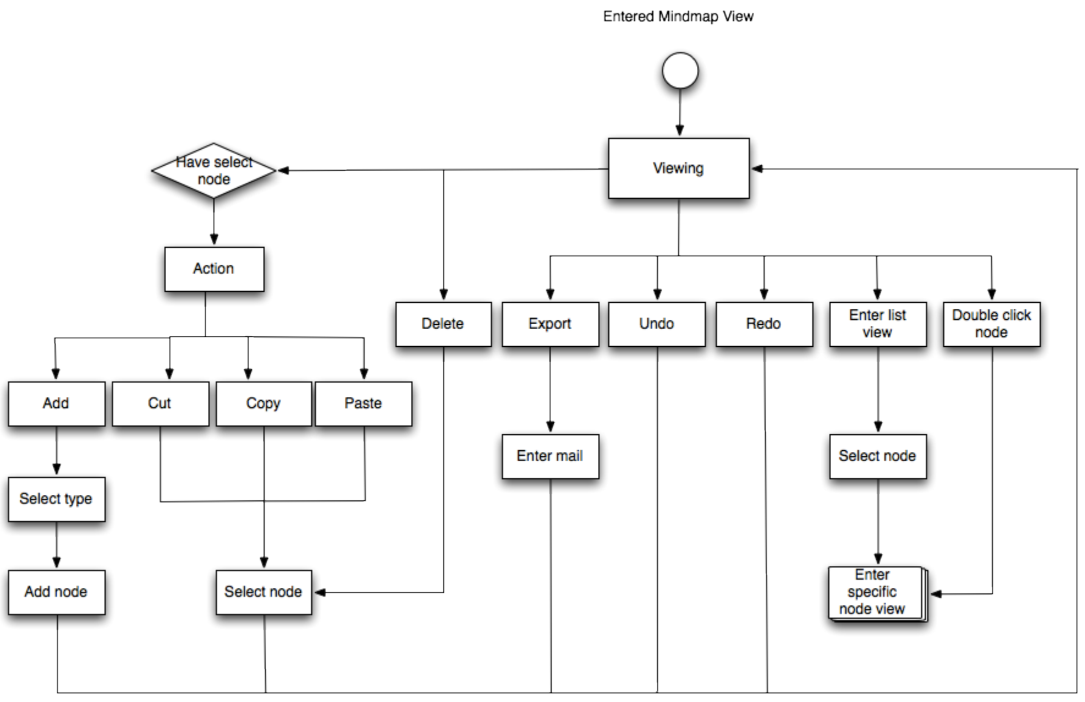
The Program Flow

Application started

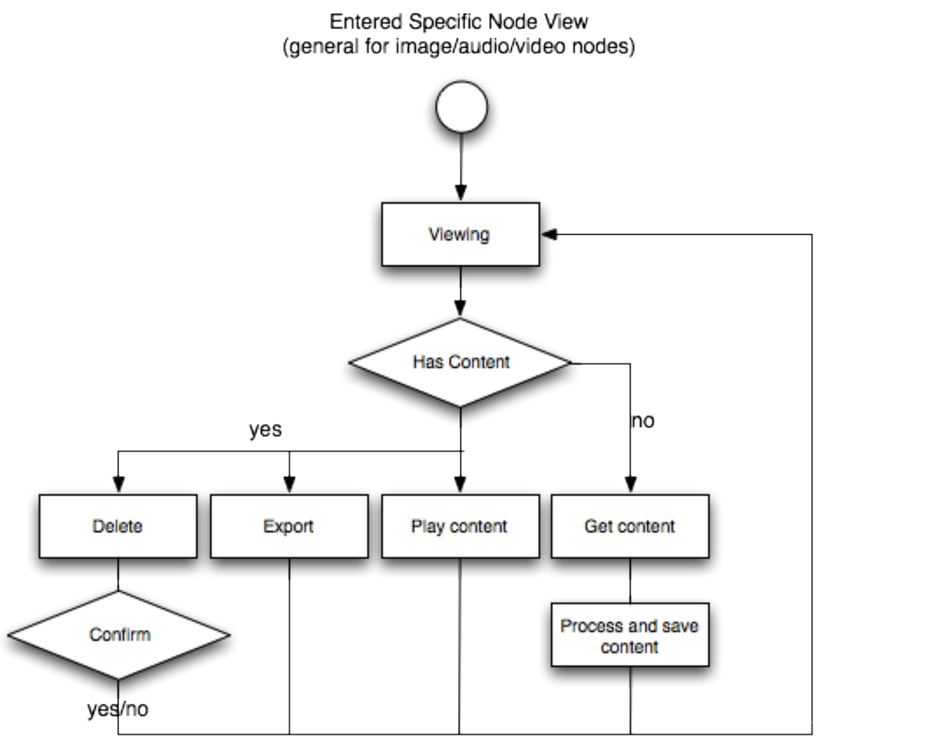


Note that in every states of this application user is allowed to exit anytime due to the nature of iPhone

The Program Flow



Note that in every states of this application user is allowed to exit anytime due to the nature of iPhone

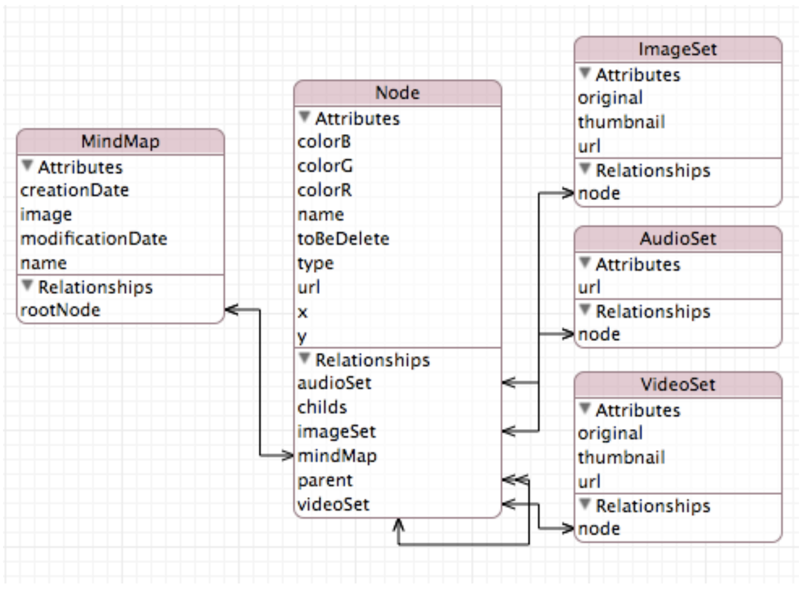


Note that in every states of this application user is allowed to exit anytime due to the nature of iPhone

DATABASE ARCHITECTURE

The structure changed from a non directional grape into a tree structure. It is because as you get more and more familiar with mind map, you will discover that mind map is basically an expansion of thoughts, from one thought to many other thoughts. Therefore the database structure is complete changed finally.

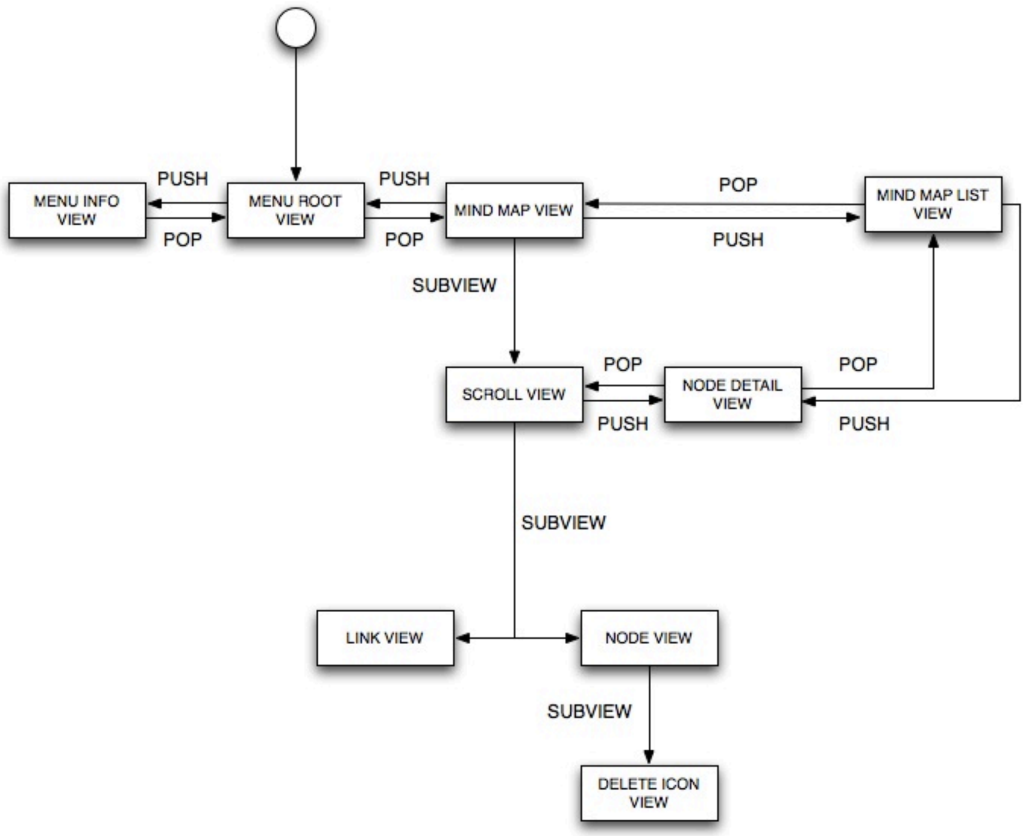
Details of the design is explained in the "Detail design" section.



VIEW HIERARCHY

Since there is no physical keyboard within iPhone, all tasks are performed within they screen. Therefore, each view is responsible have to provide different functionalities to complete different tasks. In the view hiearachy of iPhone, in order to switch tasks/view, the recommended way is to POP/PUSH view. Therefore, the application is designed base on a stack of views.

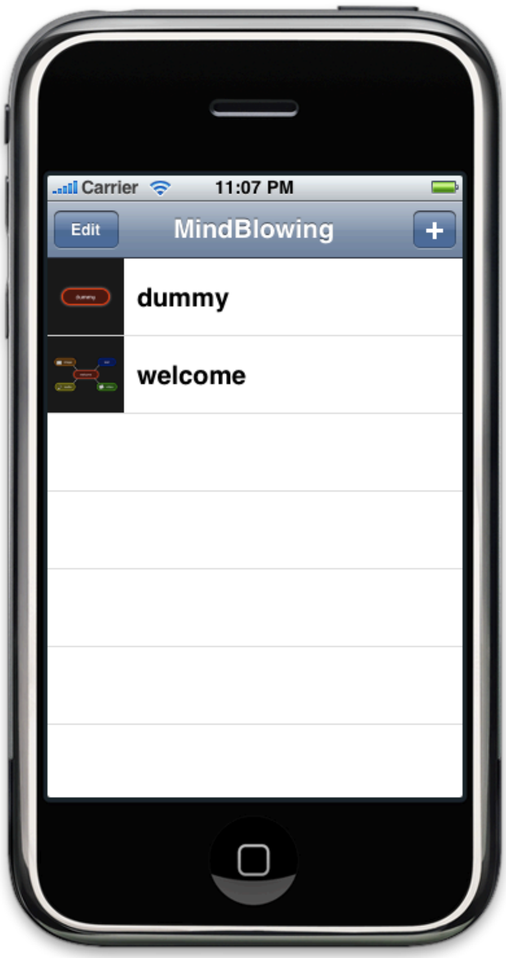
POP/PUSH – In the view hierarchy of iPhone, the views would maintain in a view stack. The parent view and its subviews would be treated as one view. In order to switch between views, the most basic and conventional view is to push the current view into stack. If you want to switch back to the last view just pop back the view. The most common usage of this is the navigation bar of views.



Due to the nature of iPhone, its entire interface is based on views. To capture the big picture of how it really works. A special view hierarchy table is provided to illustrate it. Each view has a corresponding controller in order to receive different events within that view in order to perform specific function. The functionalities and detail design and implementation are in the following sections.

MAIN VIEW CONTROLLERS

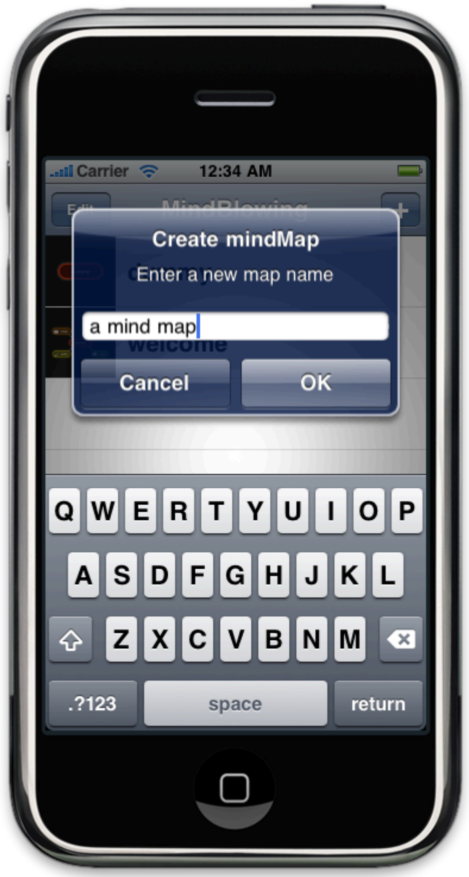
Root view controller



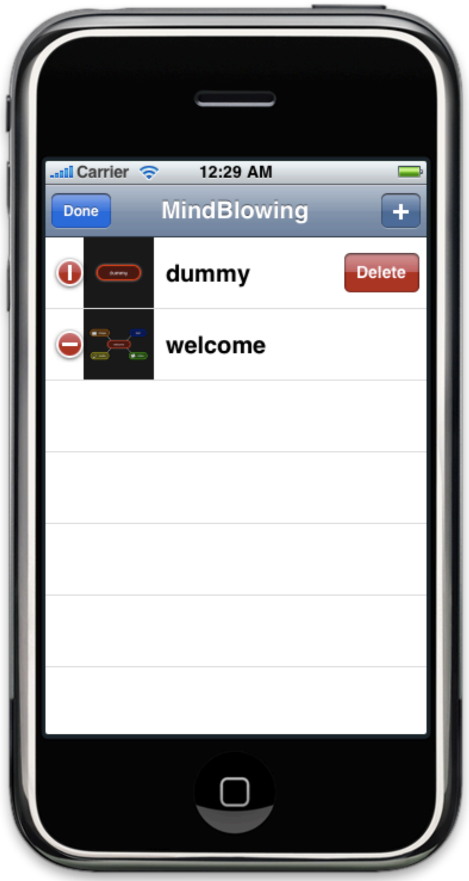
The Mindmap menu page is the first view of the application. When users wanted to start drawing or reading a Mind map, they wanted to start the application as quick as possible. The table view is the cleanest and most straight forward for user to browse through the entire Mindmap library, and in each table cell the corresponding information is displayed to the user in order to identify which Mindmap they actually wanted to access. For best user experiences, a thumbnail view of the Mindmap is in planning as well as some sorting capabilities in later development stage.

An add button is essential to be placed in within the menu view. This is kind of no brainer for users who wanted to add a Mindmap, they just simply tap the add button at the right top corner. It then brings the to an detailed view of the Mindmap and enables them to change the map name and any additional information.

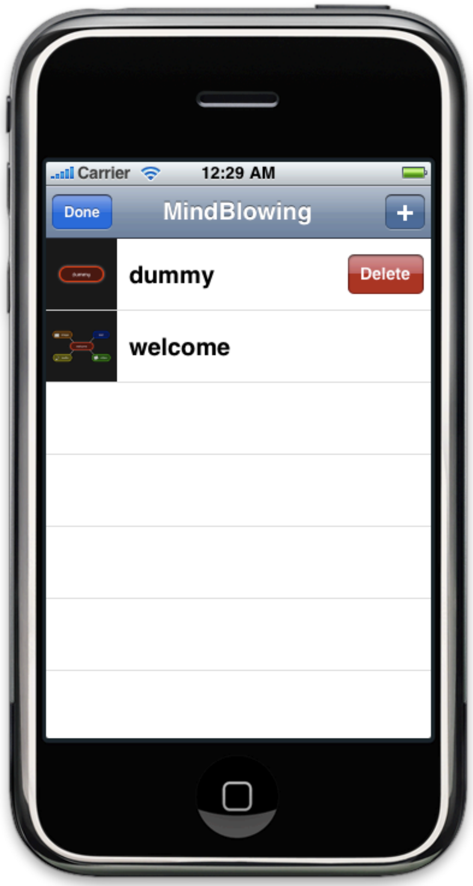
In order to delete there are two mechanism, either enter the edit mode to delete of delete each mind map individually by swifting right or left on the mind map to toggle the delete option.



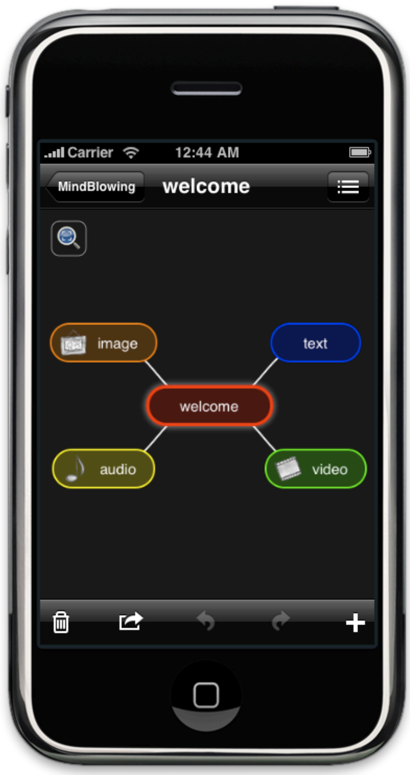
Create a mind map by pressing the + button



Toggle by pressing Edit



Toggle by swifting right/left on a mind map



Node view controller

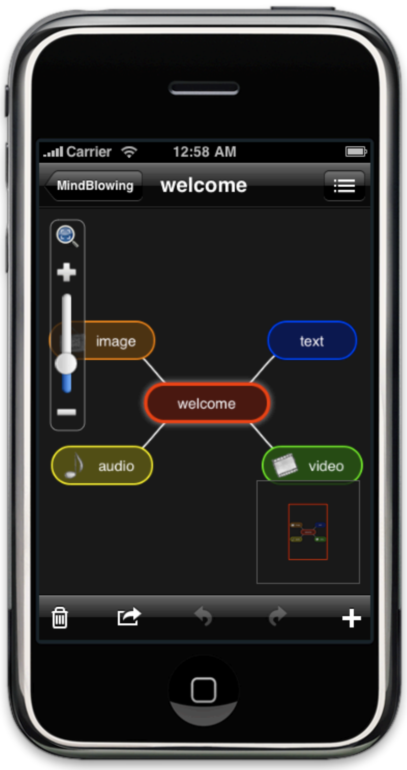
This page is the graphical presentation of the individual mind map. It provides functionalities to let user able edit the content of a mind map. The detail design and implementation are located at the following section.

All editing events are handled within this controller, which included create/delete node, undo/redo and export to mail.

This controller also contains the entries to the list view and corresponding node detail view when user triggers the correct event. The details will be explained in the detail design.

There are al together 5 types of node which are Root, Text, Image, Audio, Video. Except Root and Text only contain text as their content, the other types can also contain other media content (Audio/Video/Image)

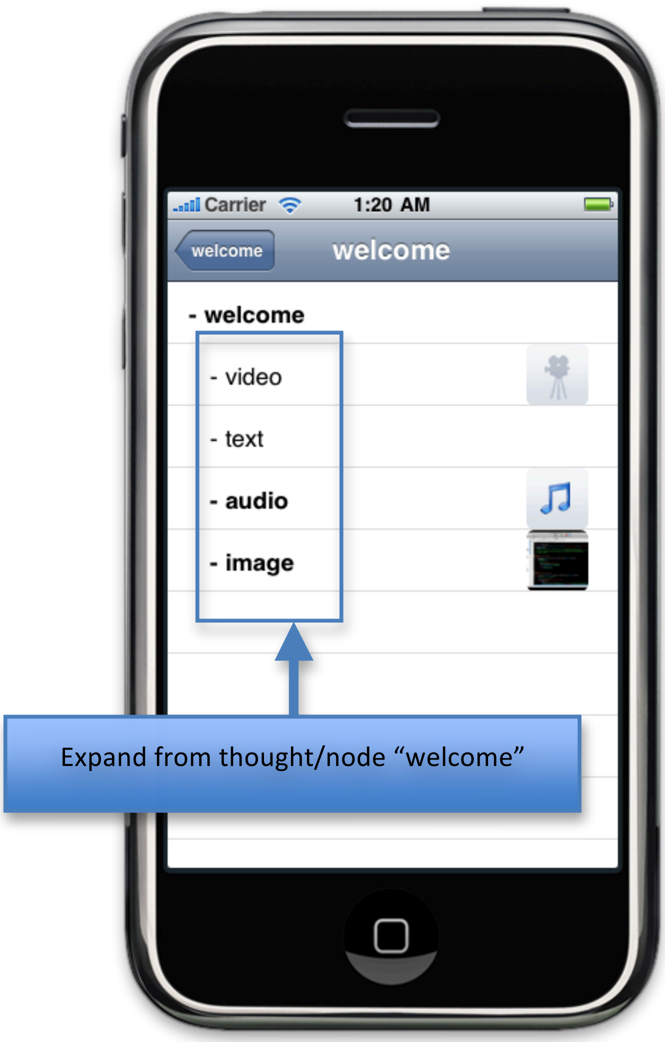
Besides the editing and navigation function, the controller also contains a navigation helper. It serves only one purpose that is to enhance the navigation within the large canvas of the mind map. There is an overview which shows the whole mind map in a relatively small portion on the screen. Besides the overview, there is a slider bar as well to control the zooming ratio.



List view controller – Tree structure

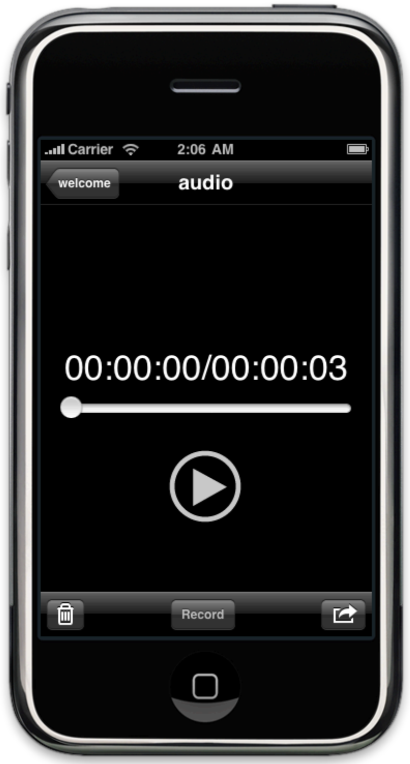
This view controller provides a tree structure of the mind map. User can browse through the structure of the mind map quickly. As the graphical representation of node view controller will become a bit clumsy when finding the thoughts/nodes expanded from particular node when there are lot of thoughts/nodes, a table view of tree structure.

User can also navigate to the detail of individual node by clicking the corresponding row of specific node.

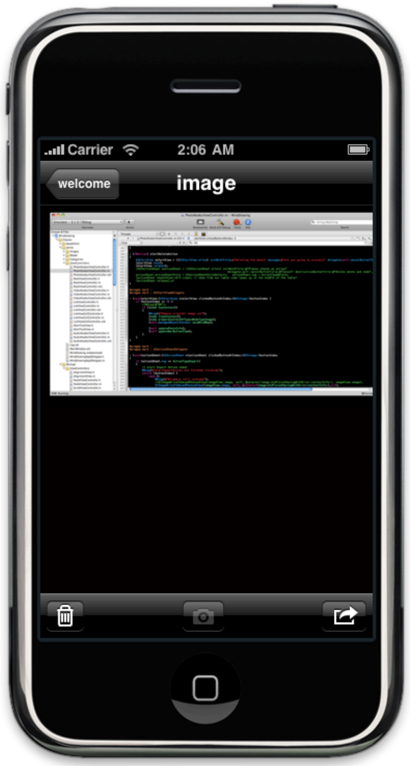


Detail node view controller

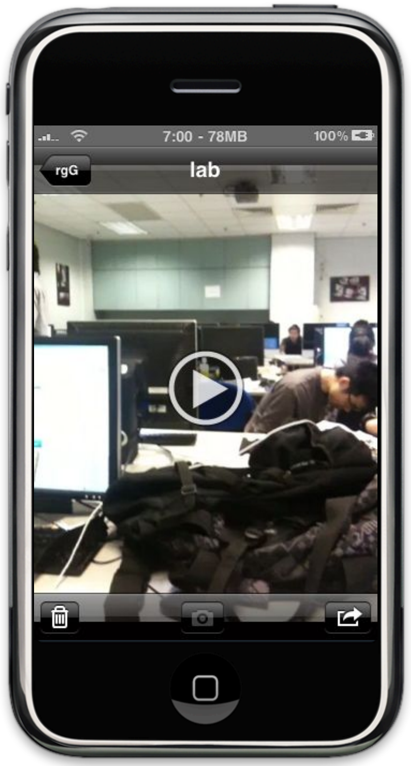
Except Root and Text node only contain text content. Image, Audio, Video can append specific media content to the corresponding node. Users can playback/review the content in this view controller. It provides decent control to different content type for user to view back the content.



Audio



Image



Video

Development Platform

OS: MAC OSX 10.6.2

Tool kits:

- Code writing: Xcode 3.2.1
- iPhone SDK 3.0 - 3.2
- Interface building: Interface Builder 3.2
- Performance turning: Instruments 2.0

Other tools:

- SVN 1.6.5
- http://beanstalkapp.com/ for svn hosting
- Dropbox

Deployment target

iPhone 3.1 & 3.1.2 simulator
iPhone 3GS + 3.1.2 firmware
iPod 1st Gen + 3.1 firmware

3. Detail Design

Justification of Users

Since my group is developing a Mindmap application, which is considered to be a notes taking and ideas elaborating helper application, I think our serious users would be students, workers, and creative content creators. I assumed they're young and interested in using productivity software for making their life easier. They probably don't need a very fancy UI, but the program should be in very good in performance and be reliable. I decided to keep the interface as clean and as comfort as possible.

What I had designed

- The application logo
- The launch screen
- The Menu
- The Database
- The Listed Tree-View
- The Photo Node View
- The Video Nodes View
- The Audio Nodes View

MENU

The first thing I've built is the menu of the Mindmap. I considered the following things when I started my design:

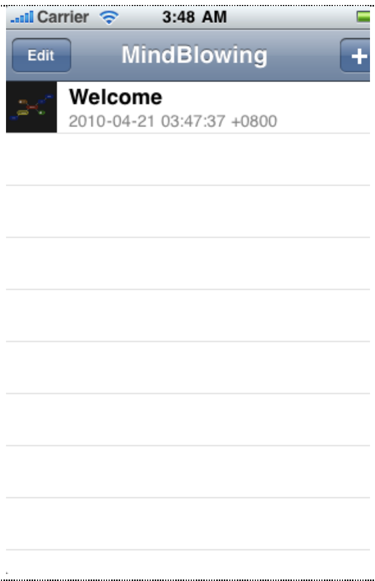
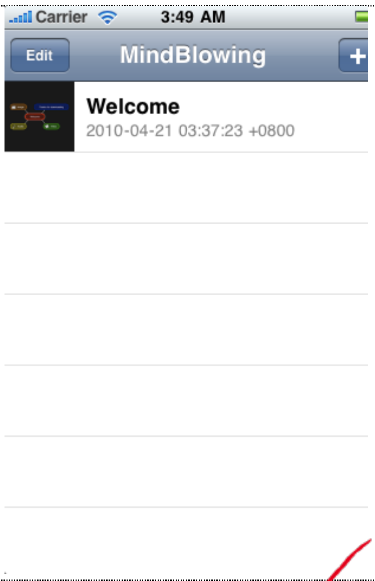
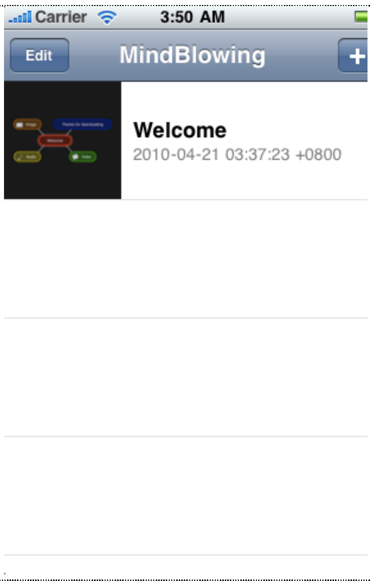
1. The application should launch fast!
2. It should be as clean as possible
3. It should enough useful information for users to be able to quickly identify which map they're using.

There're many options of a menu to be displayed. Some really fancy one like the Cover-flow one by the iPod music application, or the icons view like our desktop windows manager, and also a list view of all contents.

Since I want my application to launch as fast as possible, I tried to use the default table view as the menu for the application. The main control are placed on the top of the navigation bar

The default row height of the table view is 44px, and we think that it's definitely too small for a preview. I've tried to increase the height and it looks a lot more comfortable and better. The 100px one looks nice but having only 4 rows to be able to shown at a time would probably not enough for practical use.

I've temporary chosen the 60px to be the default row height.

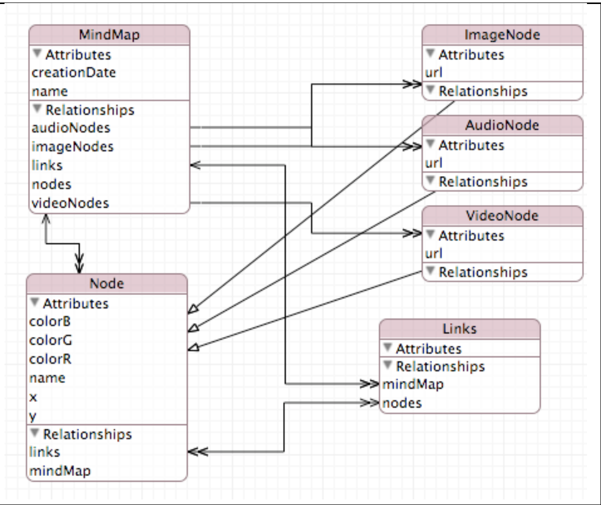
 Carrier 3:48 AM Edit MindBlowing + Welcome 2010-04-21 03:47:37 +0800	 Carrier 3:49 AM Edit MindBlowing + Welcome 2010-04-21 03:37:23 +0800	 Carrier 3:50 AM Edit MindBlowing + Welcome 2010-04-21 03:37:23 +0800
Default 44px	60px ✓	100px

THE DATABASE

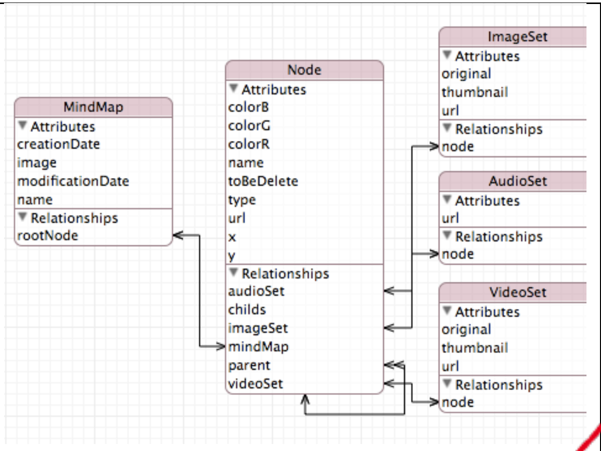
We have learnt that there're several choices for our storage format:

- ~~Simple XML file-based with custom parser~~
- ~~.plist file-based storage~~
- ~~SQLite on the iPhone OS~~
- Core Data for the iPhone OS

I've showed the reason of adopting Core Data on the last report, and I now introduced schema v2 on this semester after having deeper understanding the mechanisms of how iPhone handles different kinds of media, and also the nature of mindmap.



Schema v1 on sem1



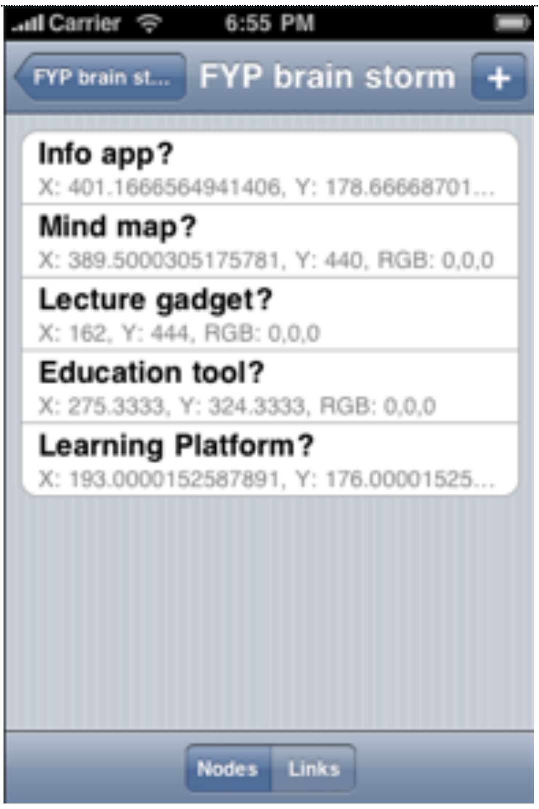
New Schema v2 on sem2

The improvements are:

- Node types are now more flexible and can be extended to store different kinds of media in a single node.
- Removed unnecessary tables and attributes.
- In order to support Undo/Redo feature of the SDK
- To support tree hierarchical of the nodes representation

The Listed Tree-View

The graphical way is a good way to present mindmap, but there would be nice to have a list view of your nodes, especially in a tree view.



Carrier 6:55 PM

FYP brain st... | FYP brain storm | +

Info app?
X: 401.1666564941406, Y: 178.66668701...

Mind map?
X: 389.5000305175781, Y: 440, RGB: 0,0,0

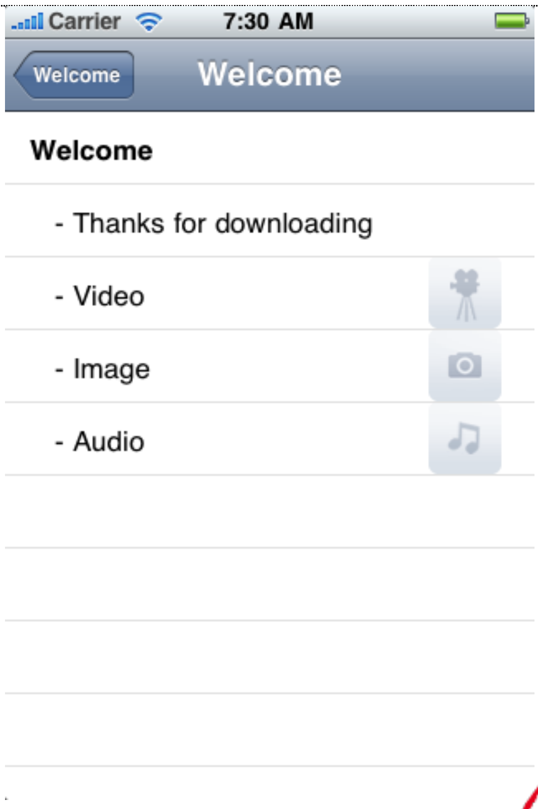
Lecture gadget?
X: 162, Y: 444, RGB: 0,0,0

Education tool?
X: 275.3333, Y: 324.3333, RGB: 0,0,0

Learning Platform?
X: 193.0000152587891, Y: 176.00001525...

Nodes | Links

Grouped list view for sem1



Carrier 7:30 AM

Welcome | Welcome

Welcome

- Thanks for downloading

- Video

- Image

- Audio

New introduced tree view

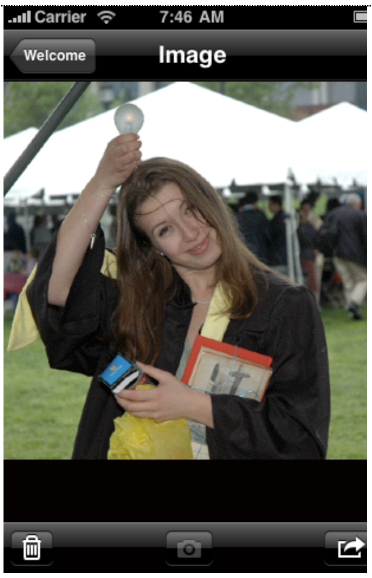
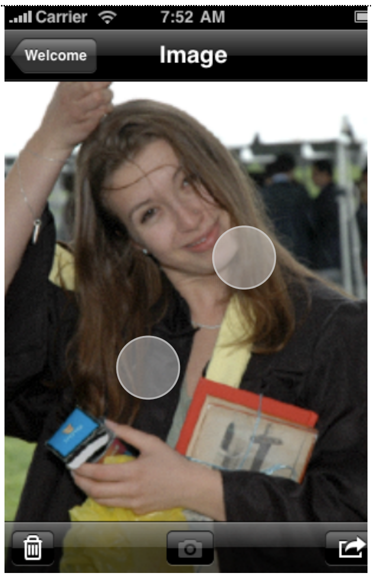
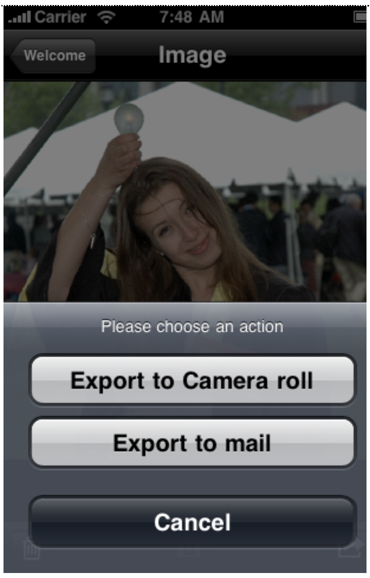
The improvements in this semester:

- Information are more organized
- Thumbnails improves users experiences
- Only needed contents are being shown

The Photo Nodes View

We have finished basic functionality on the last semester but actually not yet able to deliver media contents, but I've successfully implement it in this semester.

I wanted to make the Photo Nodes view very easy to use and support multi-touch, and here is the product.

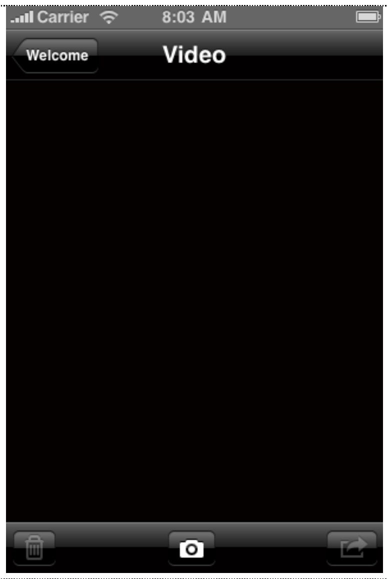
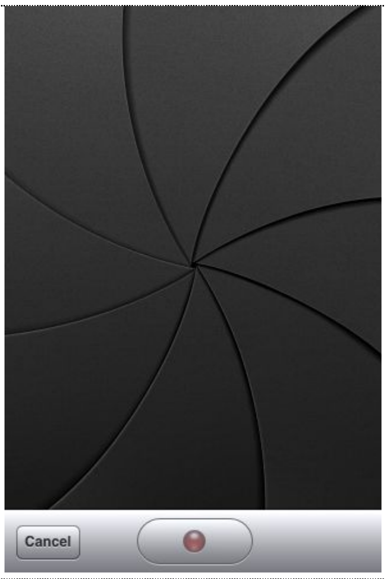
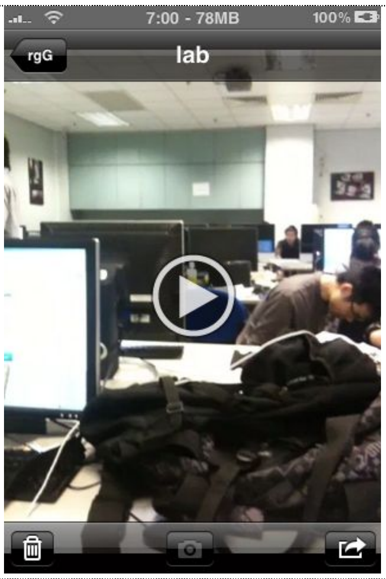
		
Loaded view	Multi-touch zooming capable	Exporting capabilities

Buttons are nicely placed in the toolbar and I've used easily recognizable icons for a fluent user experiences.

I know that some users would like to export or save their image out of the application, therefore I decided to put two options for users to share their images.

The Video Nodes View

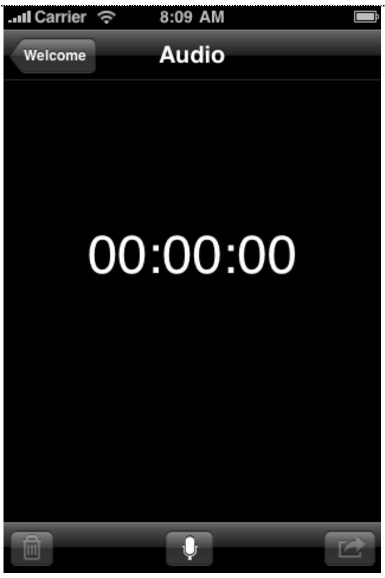
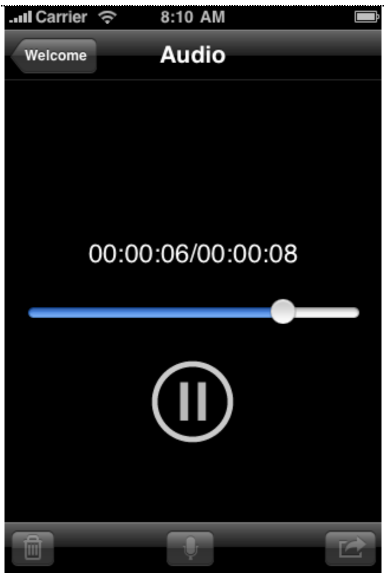
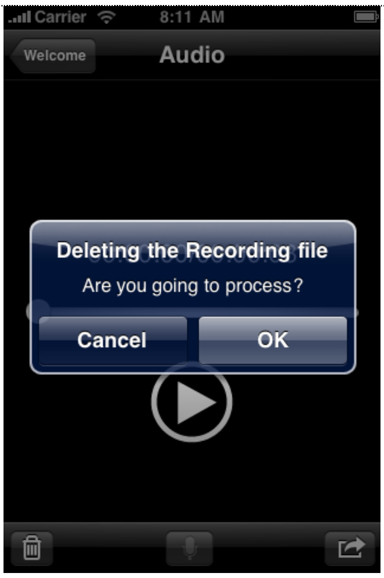
Video contents are very similar with the Audio nodes view, but instead of providing multi-touch capabilities, it should offer direct video playback and recording.

		
Loaded view without video content	Loading camera for video recording	View with video content

It also should support content exportation like the in photo nodes. The video recording module should be as similar as the integrated video capturing interface. I also decided to simulate the interface and the buttons as it is in the official Camera roll interface.

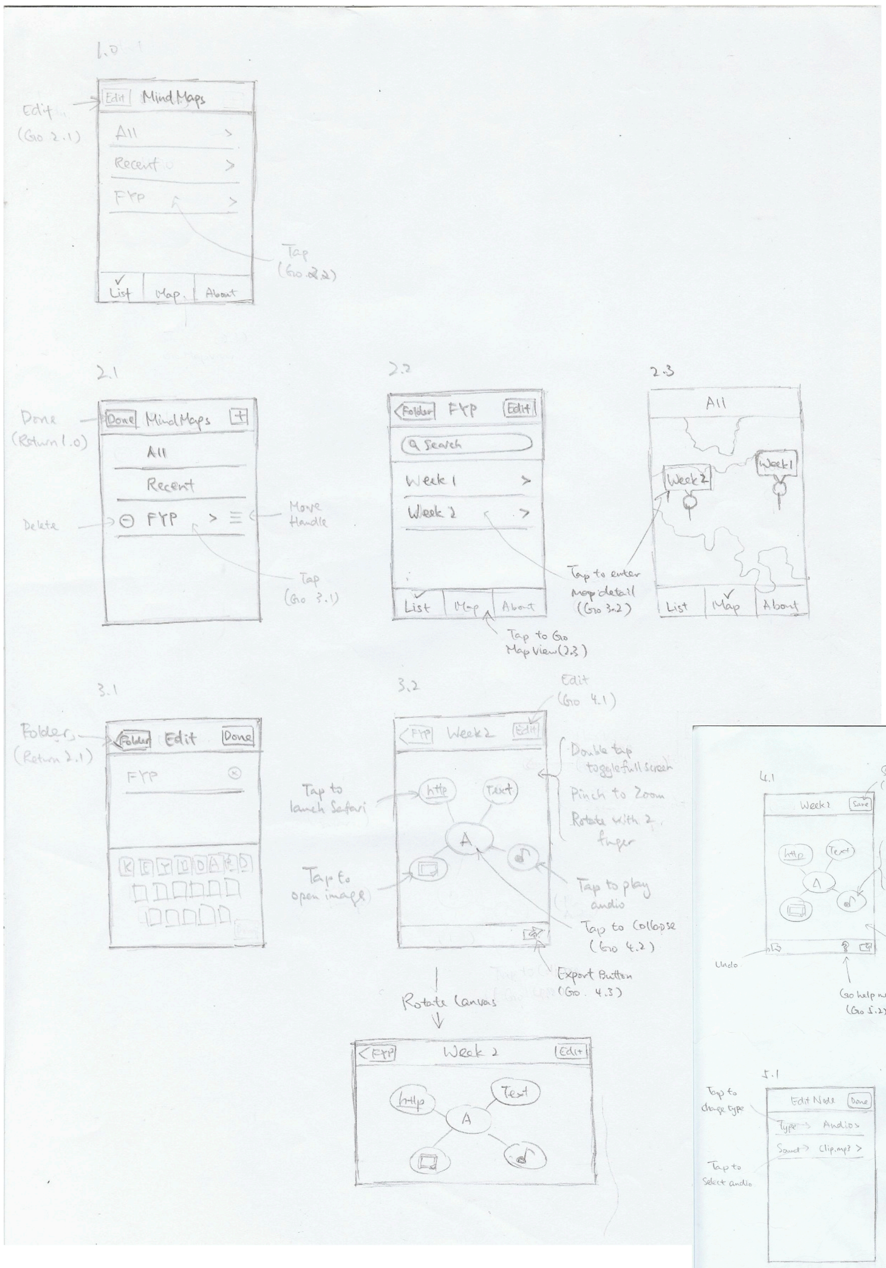
THE AUDIO NODES VIEW

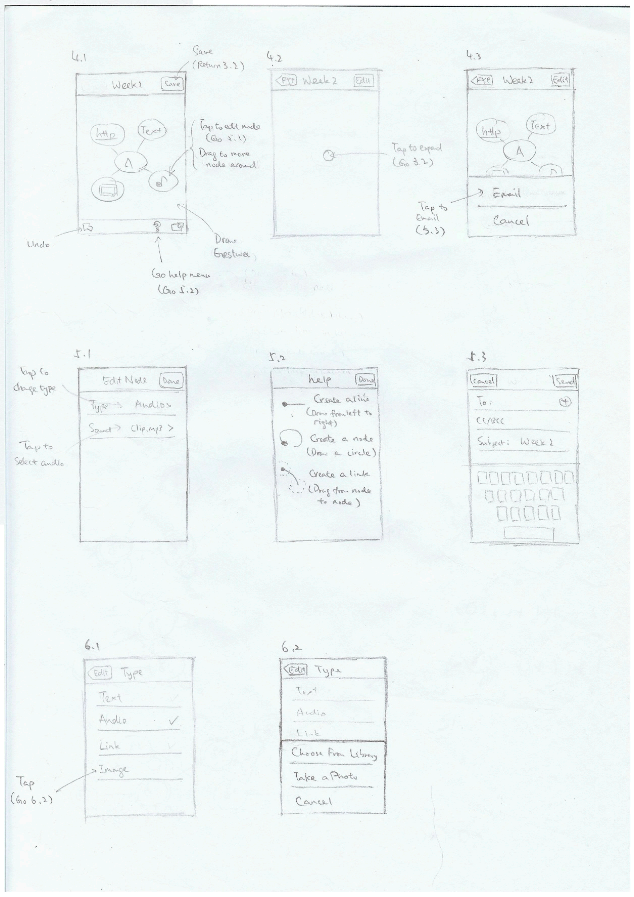
Audio contents are also very similar to the Photo and Video nodes view, and it should offer direct audio playback and recording.

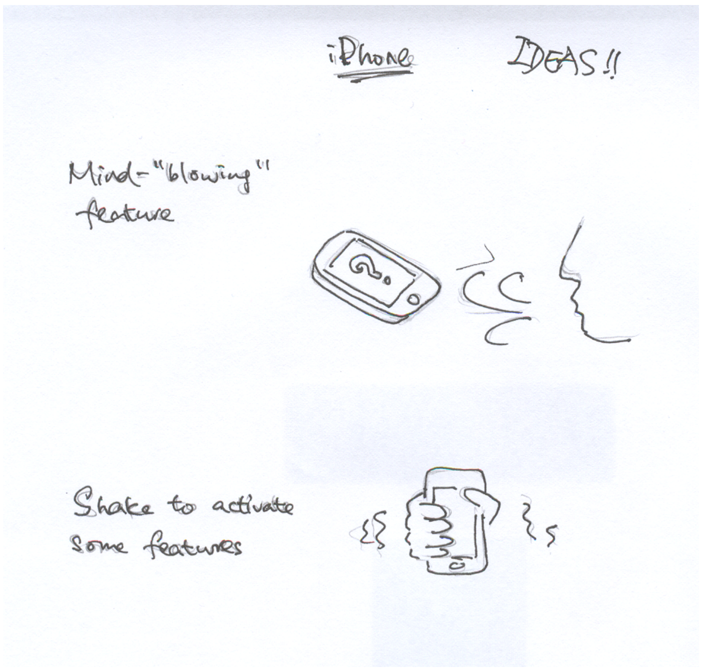
		 Deleting the Recording file Are you going to process?
Loaded view without audio content and ready for recording	Audio playback showing the progress bar and time	Prompt the user when deleting the audio file

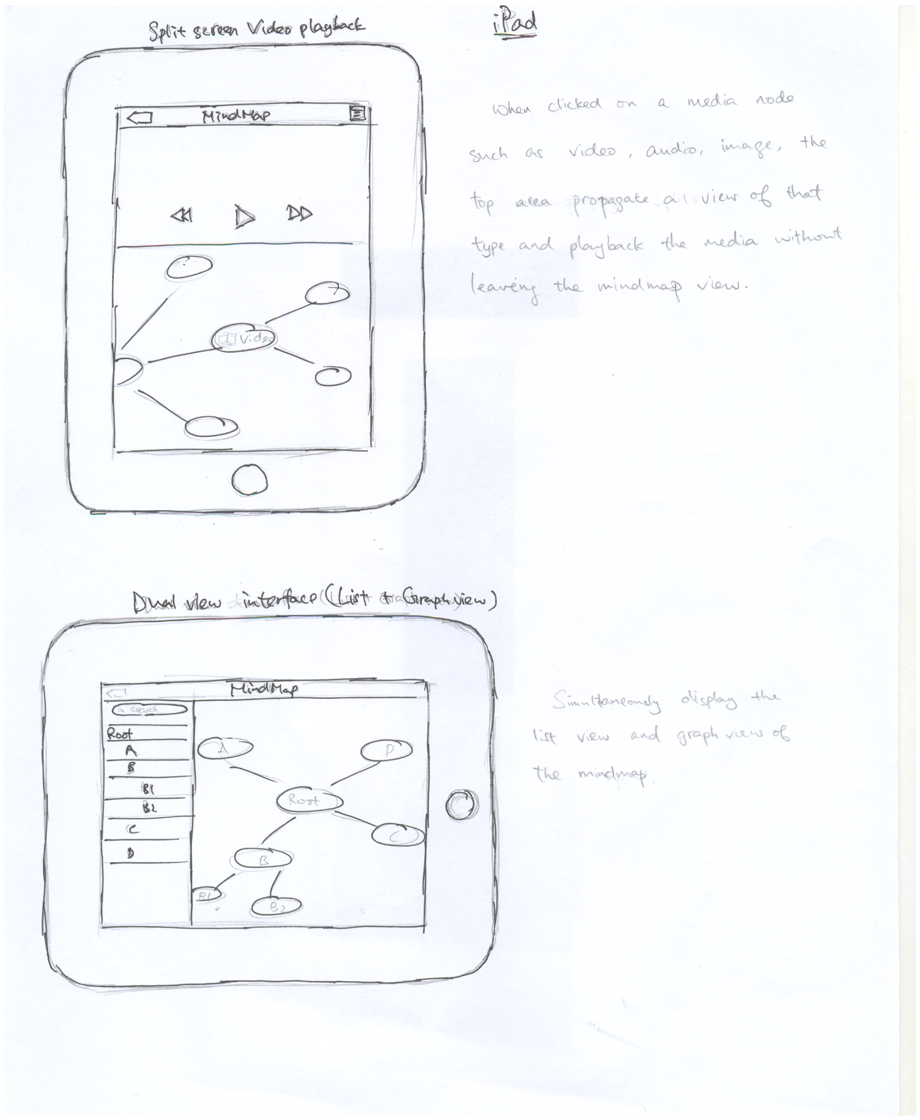
I referenced from the default music application, a progress bar can save a thousand words. All three nodes detail view should support removing content directly, and users will be prompt for before the action actually take place.

SOME INITIAL DESIGN AND IDEAS








Split screen Video playback
iPad
MindMap
Video
Dual view interface (List & Graph view)
MindMap
search
Root
A
B
B1
B2
C
D
A
P
Root
C
B
B1
B2

When clicked on a media node such as video, audio, image, the top area propagate split view of that type and playback the media without leaving the mindmap view.

Simultaneously display the list view and graph view of the mindmap.

4. IMPLEMENTATION

INITIALIZATION

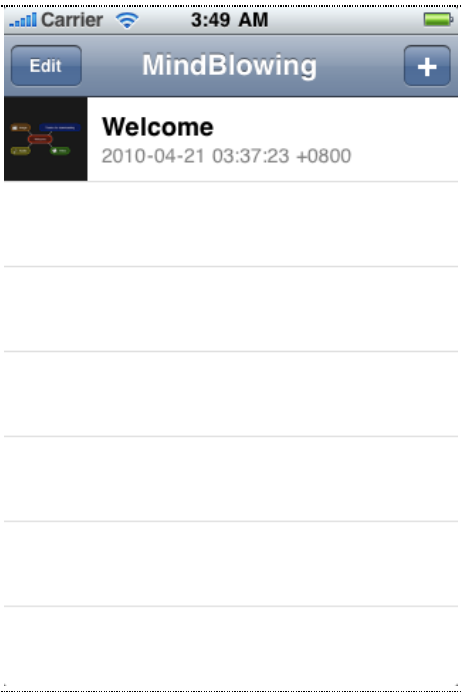
The iPhone SDK has provide excellent library for developers to use, libraries are well known as user friendly and rich. We've seen many kinds of applications and games out there. There are some very featured-rich and heavy ones, very light and useless ones, and of course there're both efficient and notable good ones. I know I am not a full-time developer and I am new to objective-C, I rather keep my application light and efficient then heavy and buggy.

In order to achieve this, I started following some of the software engineering and project management principles, as well as techniques and advanced skills of programming in objective-C.

WHAT I HAVE BEEN FOLLOWING

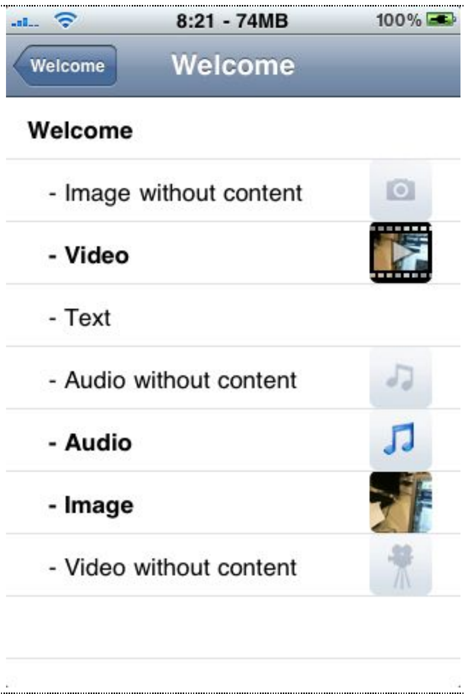
- The MVC (Model-View-Controller) approach
- The KISS (Keep it simple stupid) principle
- The Responsibility-driven design of OOP
- The DRY (Don't Repeat Yourself) principle

MENU

1. Loaded view	Runtime implementation procedure calls
 Carrier 3:49 AM Edit MindBlowing + Welcome 2010-04-21 03:37:23 +0800	1. Perform a Core Data fetch request to get the list of mindmaps. 2. Attach an event for each table view cell and let them responsible to push to the other views. 3. Attach event for the "Edit" button and "+" button. 4. Handles add event and generate a pop up for asking the name to input.

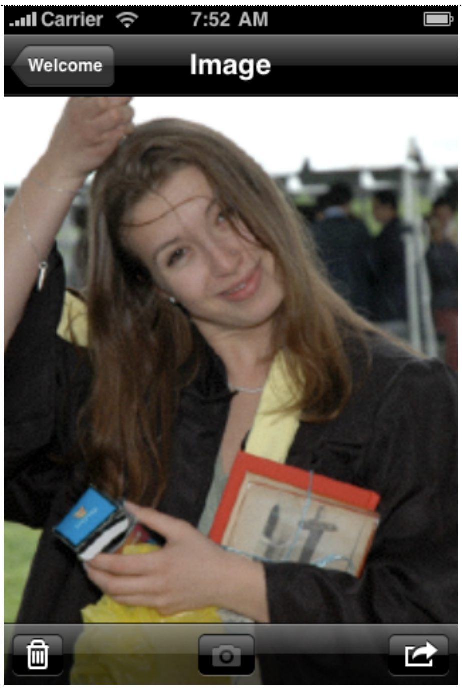
Name	RootViewController
Extends	UITableViewController
Related files	RootViewController.h RootViewController.m RootViewController.xib
Static implementation procedures	1. Create a standard UITableViewController in the Interface Builder. 2. Add two UIBarButtonItem to the top of the Navigation Bar. 3. Attach two IBAction events to handle the "edit" and "+" button.
Techniques	1. The API does not actually provide a pop up alert with text box in it. In order to achieve this result, developers have to extend the standard UIAlertView class and write their own pop up box. 2. The pop up box event can only be passed by the delegate approach to the RootViewController.

THE LISTED TREE-VIEW

1. The Listed Tree-View displaying some nodes	Runtime implementation procedure calls
 8:21 - 74MB 100% Welcome Welcome Welcome - Image without content - Video - Text - Audio without content - Audio - Image - Video without content	1. When the view is loaded, load the node array to the table view. 2. Determine the type reads for saved thumbnails or generates specific “no-content” image for the image view in the table cell. 3. Handles touches and pushes to specific type of node view controller.

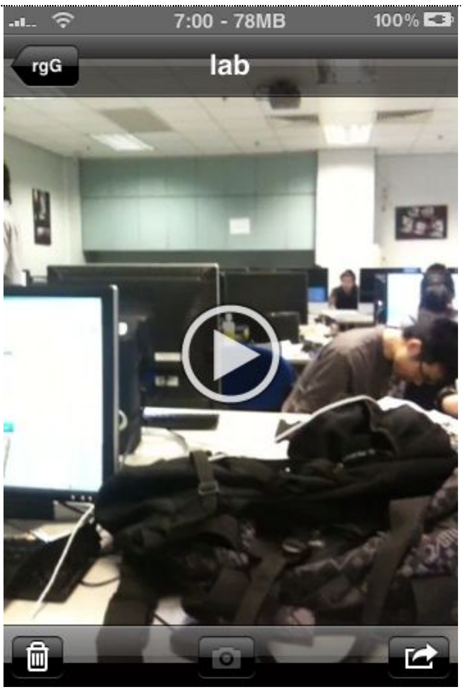
Name	ListViewController
Extends	UITableViewController
Related files	ListViewController.h ListViewController.m ListViewController.xib ListViewCellController.h ListViewCellController.m ListViewCellController.xib
Static implementation procedures	1. Create a standard UITableViewController in the Interface Builder. 2. Create a custom UITableViewCell in the Interface Builder.
Techniques	1. Thumbnails are generated by custom written code and the preview of video nodes is displayed in two different layers. 2. Standard Indentation setting will not be working in custom UITableViewCell and requires workaround for “acting“ as indentations.

THE PHOTO NODES VIEW

1. Loaded view with image content	Runtime implementation procedure calls
	1. Obtain the Node object pushed by previous controller 2. Handles touch events on the Image and pushes back to the Scroll View controller for multi-touch zooming 3. Toggle button states by determining whether image content exists. 4. Handles different button events.

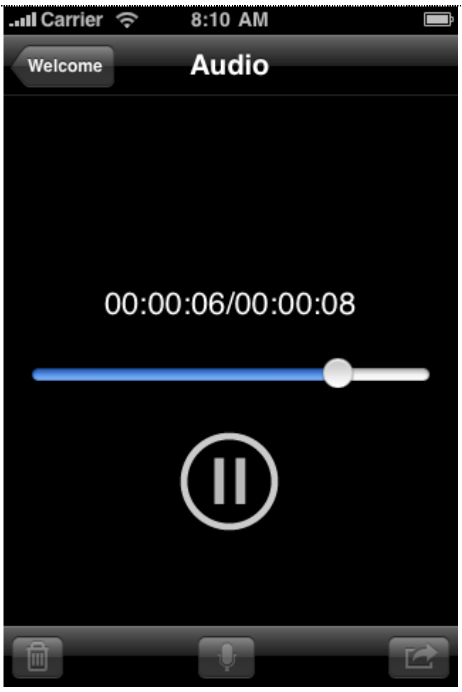
Name	PhotoNodesViewController
Extends	UIViewController
Related files	PhotoNodesViewController.h PhotoNodesViewController.m PhotoNodesViewController.xib
Static implementation procedures	1. Create a standard UIViewController in the Interface Builder. 2. Add three UIBarButtonItem to the toolbar. 3. Attach IBAction events to those buttons.
Techniques	1. Multi-touch zooming requires object delegation to pushes touch events to specific ViewControllers. 2. Saving the image to Core Data requires complex library calls to transform the image data into binary data.
Image specification	Image type: .jpg Large image: 320 (w) x 320 (h) Thumbnail: 44 (w) x 44 (h) Average photo size: ~20 kBytes

THE VIDEO NODES VIEW

1. The Loaded view with content	Runtime implementation procedure calls
	1. Obtain the Node object pushed by previous controller 2. Handles touch events on the button and toggles video playback 3. Toggle button states by determining whether image content exists, and also whether the player is stopped. 4. Handles different button events.

Name	VideoNodesViewController
Extends	PhotoNodesViewController
Related files	VideoNodesViewController.h VideoNodesViewController.m VideoNodesViewController.xib
Static implementation procedures	1. Create a standard UIViewController in the Interface Builder. 2. Add three UIBarButtonItem to the toolbar, and a button on the image view. 3. Attach IBAction events to those buttons.
Techniques	1. Make use of class extending techniques and result in less redundant and more maintainable code. 2. Generating the thumbnails of a video is quite tricky. 3. Only necessary controls are shown for different states, for example, during recording state, the play button will not appear, trash button will be disabled to prevent accident actions made by users.
Movie Specification	Movie Type: .mov Resolution: 360 x 480, 480 x 360 Average files size per second: ~100 kBytes

THE AUDIO NODES VIEW

1. Audio nodes view while audio playback	Runtime implementation procedure calls
	1. Obtain the Node object pushed by previous controller 2. Handles touch events on the button and toggles audio recording or playback 3. Toggle button states by determining whether image content exists, and also whether the player is stopped. 4. Handles different button events.

Name	AudioNodesViewController
Extends	PhotoNodesViewController
Related files	AudioNodesViewController.h AudioNodesViewController.m AudioNodesViewController.xib
Static implementation procedures	1. Create a standard UIViewController in the Interface Builder. 2. Add three UIBarButtonItem to the toolbar, a button, a slider and a label on the view. 3. Attach IBAction events to those buttons.
Techniques	1. Implementing the audio recording and playback session are totally different and more complex than it's required for photo/video playback. 2. Audio recording requires manual format settings.
Audio Specification	File type: .caf (Core Audio Format) Recording sample rate: 16000 Number of channels: 1 Depth bit rate: 16-bit Average file size per second: ~40 kBytes

5. Experiments and Evaluation

The evaluation is mainly base on testing the performance of the application CPU and Memory usage. The tool that is used to run these tests is "Instruments v2.1" which is the Apple provided tool on testing program performance that is written for Mac OS.

Instruments

Version 2.1 (1208)

Copyright © 2006–2009 Apple Inc.
All Rights Reserved.

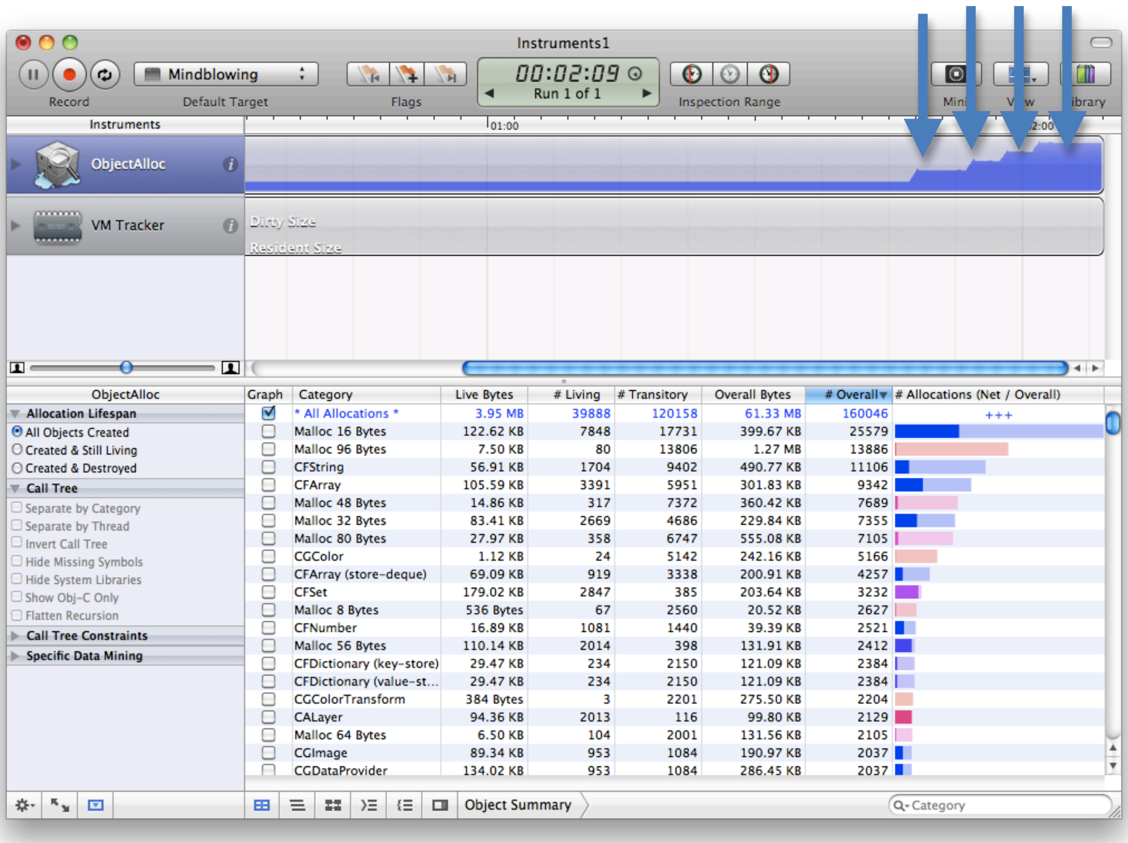
Memory test

Object allocation

creation of > 40 nodes

Object allocation only record the allocated memory, which will not minus the dealloced memory. In order to see the memory usage of the test, there is a mind map with more than 40 nodes which is created before running the test. Every reloads of that mind map indicates a rough memory allocation of the worst scenario.

Reload times: 4;



Graph	Category	Live Bytes	# Living	# Transitory	Overall Bytes	# Overall	# Allocations (Net / Overall)
☑	* All Allocations *	3.95 MB	39888	120158	61.33 MB	160046	+++
☐	Malloc 16 Bytes	122.62 KB	7848	17731	399.67 KB	25579	
☐	Malloc 96 Bytes	7.50 KB	80	13806	1.27 MB	13886	
☐	CFString	56.91 KB	1704	9402	490.77 KB	11106	
☐	CFArray	105.59 KB	3391	5951	301.83 KB	9342	
☐	Malloc 48 Bytes	14.86 KB	317	7372	360.42 KB	7689	
☐	Malloc 32 Bytes	83.41 KB	2669	4686	229.84 KB	7355	
☐	Malloc 80 Bytes	27.97 KB	358	6747	555.08 KB	7105	
☐	CGColor	1.12 KB	24	5142	242.16 KB	5166	
☐	CFArray (store-deque)	69.09 KB	919	3338	200.91 KB	4257	
☐	CFSet	179.02 KB	2847	385	203.64 KB	3232	
☐	Malloc 8 Bytes	536 Bytes	67	2560	20.52 KB	2627	
☐	CFNumber	16.89 KB	1081	1440	39.39 KB	2521	
☐	Malloc 56 Bytes	110.14 KB	2014	398	131.91 KB	2412	
☐	CFDictionary (key-store)	29.47 KB	234	2150	121.09 KB	2384	
☐	CFDictionary (value-st…	29.47 KB	234	2150	121.09 KB	2384	
☐	CGColorTransform	384 Bytes	3	2201	275.50 KB	2204	
☐	CALayer	94.36 KB	2013	116	99.80 KB	2129	
☐	Malloc 64 Bytes	6.50 KB	104	2001	131.56 KB	2105	
☐	CGImage	89.34 KB	953	1084	190.97 KB	2037	
☐	CGDataProvider	134.02 KB	953	1084	286.45 KB	2037	

Before entering the mind map, the memory allocation remains still at 2MB at the menu. Every reload increase approximately 15MB. In term of the memory in iPhone, 15MB is relatively small amount of memory usage even it is the worst case.

MEMORY LEAKS

Focus on mind map node manipulation

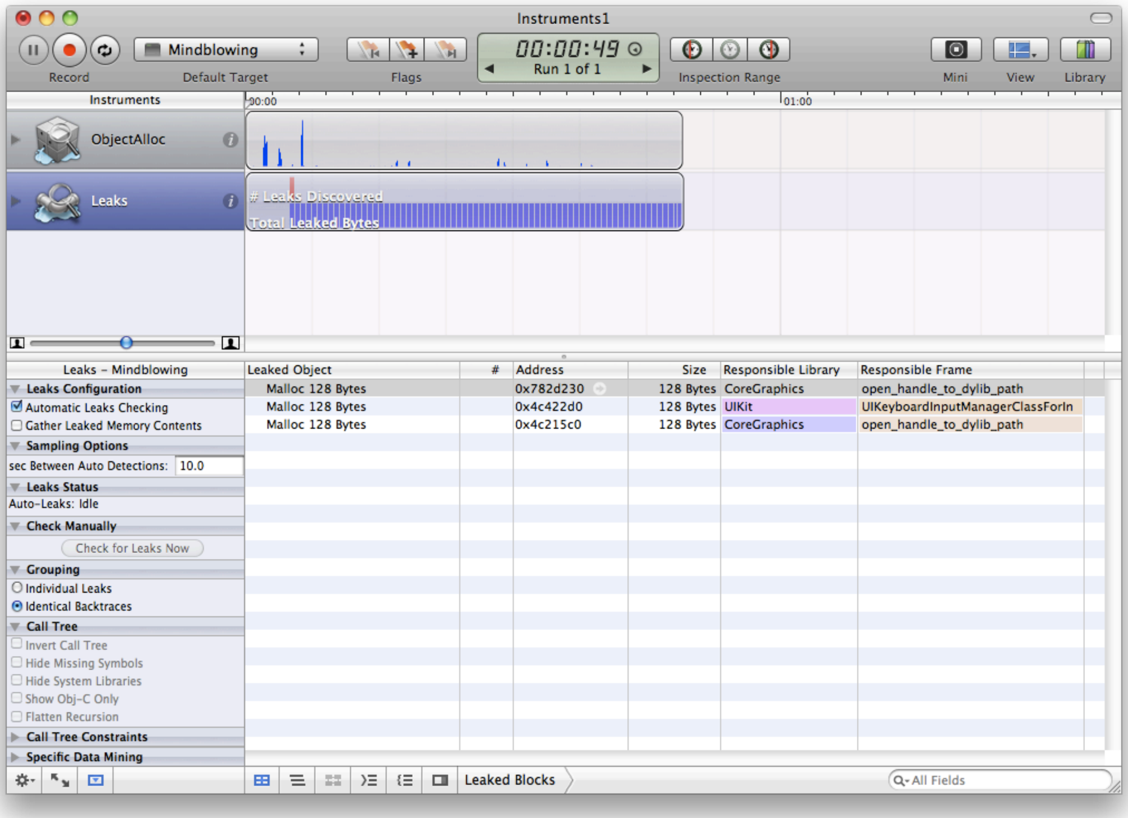
As most of the usage of the application is node manipulation, a set of actions was chosen to test during manipulating the nodes. The action sequence is listed below:

Enter a mind map

Perform the following actions at least one time in random order:

- Create all 4 types of node
- Delete node
- Drag node
- Undo + redo
- Cut + Copy + Paste

Back to menu



Leaks – Mindblowing

Leaks Configuration
- [✓] Automatic Leaks Checking
- [ ] Gather Leaked Memory Contents

Sampling Options
- sec Between Auto Detections: 10.0

Leaks Status
- Auto-Leaks: Idle

Check Manually

Grouping
- ○ Individual Leaks
- ● Identical Backtraces

Call Tree
- [ ] Invert Call Tree
- [ ] Hide Missing Symbols
- [ ] Hide System Libraries
- [ ] Show Obj-C Only
- [ ] Flatten Recursion

Call Tree Constraints

Specific Data Mining

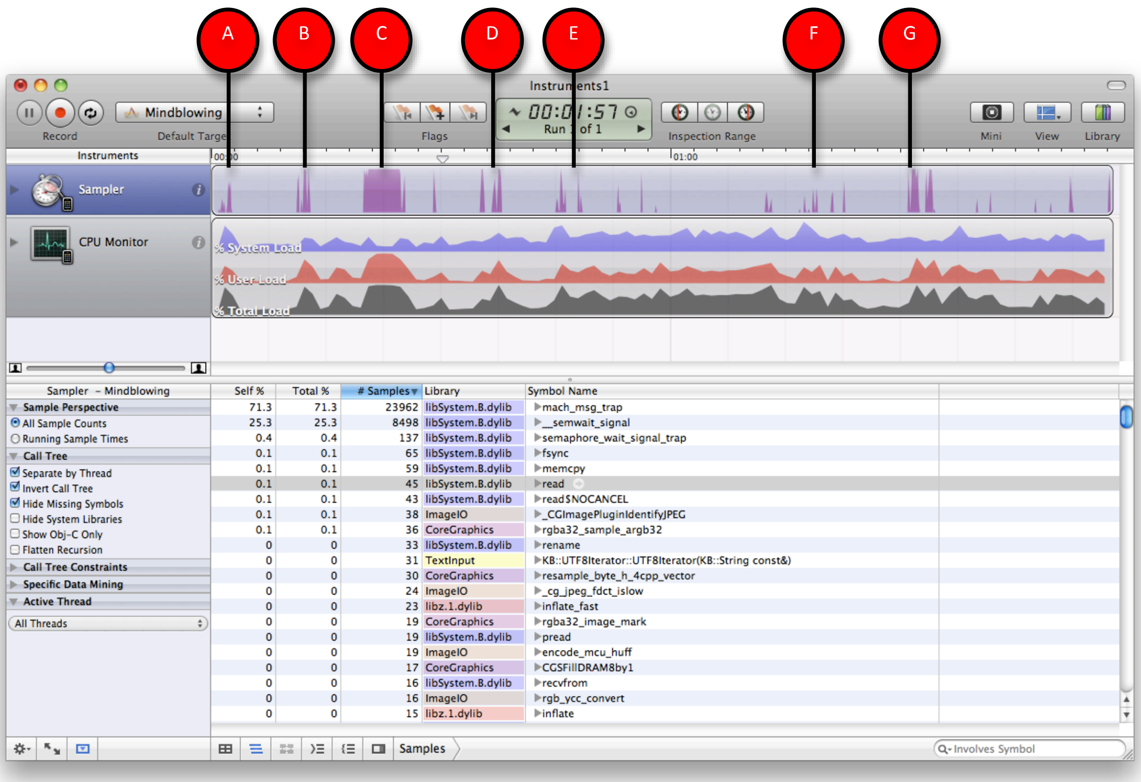
Leaked Object	#	Address	Size	Responsible Library	Responsible Frame
Malloc 128 Bytes		0x782d230	128 Bytes	CoreGraphics	open_handle_to_dylib_path
Malloc 128 Bytes		0x4c422d0	128 Bytes	UIKit	UIKeyboardInputManagerClassForIn
Malloc 128 Bytes		0x4c215c0	128 Bytes	CoreGraphics	open_handle_to_dylib_path

The testing result is dump screen as above, as you can see, there are totally 3 leakages. But the responsible library and frame have nothing to do with our program function. After a series of research, it is confirmed that Instruments simply mistakenly determine the reserved memory of the core library function as memory leak. In the test case above is the dismiss of the system Keyboard and some unknown reserved memory which is handled by CoreGraphics.

CPU Sampler

The most CPU intense time

The common actions and procedures are being captured and sampled in the following test.



Sampler – Mindblowing

Sample Perspective
- (●) All Sample Counts
- ( ) Running Sample Times

Call Tree
- ☑ Separate by Thread
- ☑ Invert Call Tree
- ☑ Hide Missing Symbols
- ☐ Hide System Libraries
- ☐ Show Obj-C Only
- ☐ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread: All Threads

Self %	Total %	# Samples	Library	Symbol Name
71.3	71.3	23962	libSystem.B.dylib	mach_msg_trap
25.3	25.3	8498	libSystem.B.dylib	__semwait_signal
0.4	0.4	137	libSystem.B.dylib	semaphore_wait_signal_trap
0.1	0.1	65	libSystem.B.dylib	fsync
0.1	0.1	59	libSystem.B.dylib	memcpy
0.1	0.1	45	libSystem.B.dylib	read
0.1	0.1	43	libSystem.B.dylib	read$NOCANCEL
0.1	0.1	38	ImageIO	_CGImagePluginIdentifyJPEG
0.1	0.1	36	CoreGraphics	rgba32_sample_argb32
0	0	33	libSystem.B.dylib	rename
0	0	31	TextInput	KB::UTF8Iterator::UTF8Iterator(KB::String const&)
0	0	30	CoreGraphics	resample_byte_h_4cpp_vector
0	0	24	ImageIO	_cg_jpeg_fdct_islow
0	0	23	libz.1.dylib	inflate_fast
0	0	19	CoreGraphics	rgba32_image_mark
0	0	19	libSystem.B.dylib	pread
0	0	19	ImageIO	encode_mcu_huff
0	0	17	CoreGraphics	CGSFillDRAM8by1
0	0	16	libSystem.B.dylib	recvfrom
0	0	16	ImageIO	rgb_ycc_convert
0	0	15	libz.1.dylib	inflate

A – loaded into the application
B – Enter a mindmap with 20 nodes
C – Export a mindmap to mail with 20 nodes
D – Playing a video
E – Going into list view with 20 nodes
F – switching between different view controllers
G – Initialize an audio recording

As we can see in the result, the heaviest loading for the iPhone is the time exporting contents to the mail application. It actually results in a tiny non-responsive period, but we can imagine this is not an often operations for users and it will not cause a large impact for them.

The results showed that the phone is able to handle other media playback quite well and the performance is satisfied when especially when using higher specification model such as the iPhone 3GS used in this test.

We believed that the shipping iPad and upcoming iPhone 4G can run our program more smoothly.

6. Conclusion and Future work

Project outputs and achievement

As it turns out, our group has achieved to create a fully function mind map application with extra functionalities that utilized the unique features of iPhone as well as enhance the usability of mind map on iPhone. The full list of completed features is listed in the "Recap" section, the comparison of features achieved between semester 1 and 2.

Most of the effort is put on testing, refining and exception handling in this semester. Although semester 1 there was already a prototype of the application, it is just a skeleton of the mind map application. Most of the advanced features were not yet done and the program was full of bug and exception. However, in semester 2, our group is managed to create a rather robust application that provide enhanced features that other similar applications do not provide. For the comparison of features please refer to the "Comparison" section.

At the moment of submitting this report, the application is being process to submit to app store (Only Team Agests, our System Admin. in this case, can prepare and submit application to app store according to the instructed regulation) or even in the middle of approval process.

Future work

There are still a few features in the "May be nice to included" category.

- Auto resizing canvas
- Auto rearrangement of node position
- Canvas setting
- Different node style
- Expand/collapse node

The above features are some potential features but due to the limited time frame and relatively less impact on the output, they were put into pending at the very beginning of semester 2.

Moreover, the GUI is a "never end" refinement process especially when you can even get back feed back from user.

Self-reflection

Getting participated in the development of an iPhone application has really taught me a lot. From the start of design to the process of implementation, I think I've discovered the art of coding. I learn and applied new programming techniques in this project and spend a large portion of time to rethink and reconstruct my code to become cleaner and more reusable. Sometimes it looks like the effort is nowhere noticeable and very time consuming, but since the finalization of our implementation, I feel that my effort has been paid off.