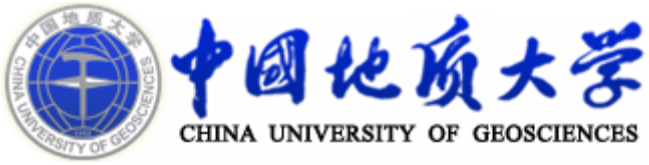

《嵌入式开发技术》

项目报告

姓　　名：	门一凡
学　　号：	1004146124
学　　院：	信 息 工 程 学 院
专　　业：	软件工程
指导教师：	龙腾

目录

一、功能说明

1. 加载当日必应壁纸

本 App 可以加载当天的 Bing 主页壁纸，当用户每天打开 App 时，App 主页中的壁纸即为当天的壁纸。（图 1.1）

2. 访问历史壁纸

可以通过日期的加减访问过去 Bing 主页壁纸的功能，由于图片存储在本人服务器中，最早可以访问到 2016 年 9 月 28 日的壁纸。（图 1.2）

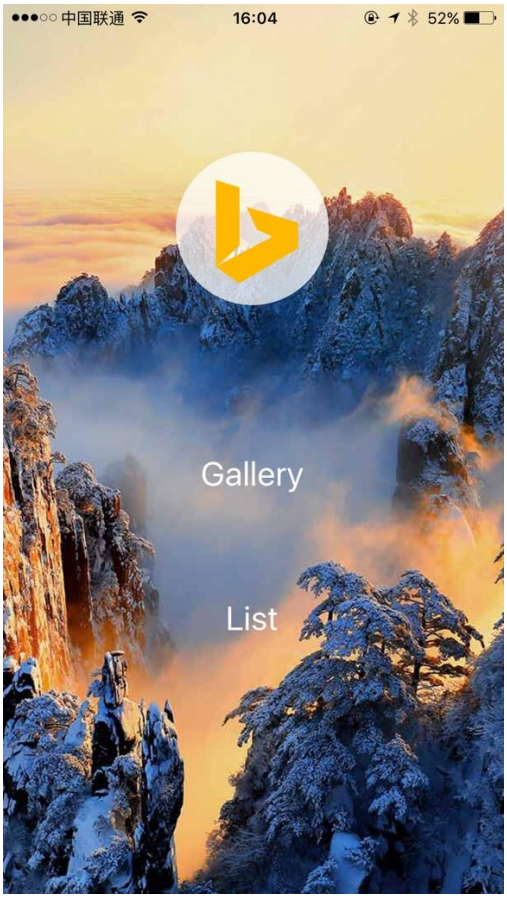

图 1.1

图 1.2

3. 列表式浏览历史壁纸

对于所有浏览过的壁纸，本 App 可以生成列表，并且按照月份划分，同时按照日期倒序排列，用户可以直接找到自己访问过的壁纸，对于未访问过的壁纸则不会显示，减小流量消耗。（图 1.3）

4. 保存壁纸到本地

提供一键保存到本地功能，用户点击后即可在 iOS 的图库 App 中找到刚才保存的图片，图片的分辨率无压缩，为 1920*1080p，适合用作壁纸。（图 1.4）

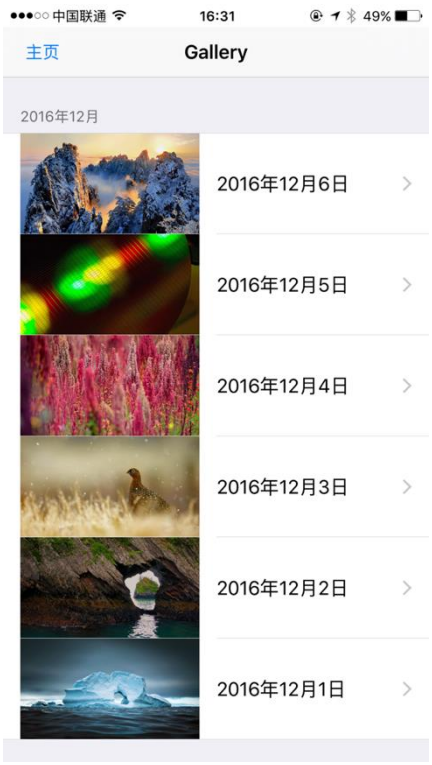

图 1.3

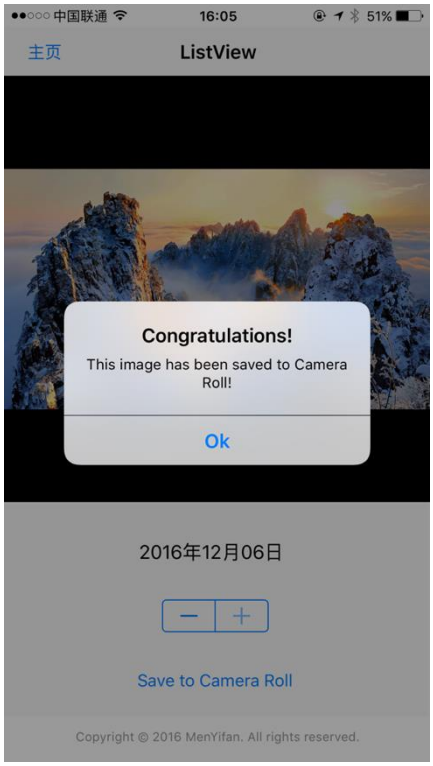

图 1.4

5. 大小图预览切换

在 ListView 界面中，用户可以轻触图片，达到切换大小的功能。

图 1.5

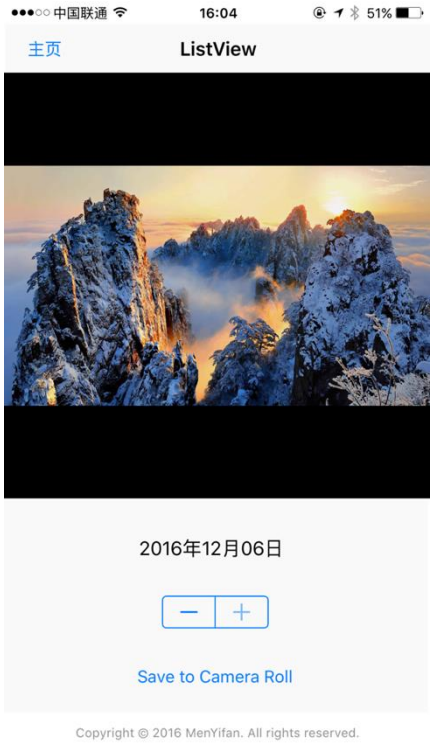

图 1.6

二、界面设计说明

1. ViewController 类

ViewController 继承了 UIViewController 类，对应一进入 App 时的场景，这个类控制主场景。

这个场景加载时会自动判断今日的壁纸是否下载到本地，如果下载到本地，则可以直接加载，否则下载后再加载。

(1) DateFormatter 类对象：dateFormatter2

这个对象用于格式化日期，用来使 Date 类型转化为获取图片的 URL 所需要的"yyMMdd"格式。

(2) viewDidLoad 函数

这个函数重载了系统自带的函数，在这个函数中，程序将获取今天的日期，判断本地是否有今天的图片，如果有的话则直接加载，没有的话则调用 downloadImage 函数实现下载并加载的功能。

(3) getDataFromUrl 函数

从 Url 中获取数据

(4) downloadImage 函数

调用 getDataFromUrl 函数得到数据，将数据赋值给 data 变量。在得到图片后将场景中的图片设为从网络上获取的图片。同时下载到本地 Document 目录。

(5) getDirectoryPath 函数

返回当前 App 的 Document 目录

2. Gallery 类

Gallery 类继承了三个类，分别是：UIViewController, UITableViewDataSource, UITableViewDelegate。

Gallery 类对应点击 Gallery 所进入的场景，这个场景包含了一个列表式的浏览器即一个 UITableView 类的控件，App 会根据本地已有图片生成列表，并且按照月份划分同时按照时间倒序排列，为用户提供了较好的使用体验。

(1) DateFormatter 类型变量：dateFormatter2

这个对象用于格式化日期，用来使 Date 类型转化为 Document 目录中的文件所需要的"yyMMdd"格式。

(2) String 数组类型变量：pic

用于记录所有图片名称

(3) Int 数组类型变量：months

对于每一个图片，程序中都是用 6 位数的 ID 表示，那么前 4 位则用于按照

月份划分。

(4) 从 int 指向 int 数组的 dict 类型变量：dic

对于每一个月份特征值要记录他对应了哪些 id。

(5) Date 类型变量 currentDate

用于记录所选项所对应的时间，用于变量传递。

(6) viewDidLoad 函数

重载了自带的函数，在这个函数中，程序初始化了所有变量，按日访问所有可以访问的日期图片，并且对于变量进行赋值

(7) numberOfSections(in tableView: UITableView) -> Int

对于 ViewController 中的 UITableView，设置栏目的数量

(8) tableView(_ tableView: UITableView, numberOfRowsInSection section:Int) -> Int

对于 ViewController 中的 UITableView，设置每个栏目中的条目数量。

(9) tableView(_ tableView: UITableView, titleForHeaderInSection section:Int) -> String?

对于 ViewController 中的 UITableView，设置每个栏目的标题。

(10) tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell

对于 ViewController 中的 UITableView，设置每一个条目的图片以及文字信息。

(11) tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath)

对于 ViewController 中的 UITableView，设置每一个条目被点击后发生的事件。

(12) getDirectoryPath 函数

返回当前 App 中 Document 目录的路径。

3. ListView 类

ListView 继承了三个类：UIImagePickerControllerDelegate, UIViewController, UINavigationControllerDelegate。

ListView 负责 List 场景中的内容，主要包括通过一个 Stepper 对于当前图片的改变。

(1) DateFormatter 类型变量 dateFormatter 和 dateFormatter2

分别用于格式化日期，前者用于格式化 Label 中文字的格式，后者用于格式

化获取链接时的格式。

(2) String 类型变量 imageURLString

用于记录图片的链接网址。

(3) Date 类型变量 today 和 current 和 startDate

today 和 current 分别用于记录今天的日期和当前图片所对应的日期，startDate 用于记录图片开始的日期，这里 startDate 固定为 2016 年 9 月 28 日。

(4) Bool 类型变量 flag

用于记录是否是由 Gallery 类跳转过来，如果是的话则为 false，否则为 true。

(5) viewDidLoad 函数

重载自带的函数，初始化两个日期格式器、图片预览形式以及 Stepper 的属性。

(6) getDirectoryPath() -> String 函数

返回当前 App 中 Document 目录的路径。

(7) getDataFromUrl 函数

从 Url 中获取数据

(8) downloadImage 函数

调用 getDataFromUrl 函数得到数据，将数据赋值给 data 变量。在得到图片后将场景中的图片设为从网络上获取的图片。同时下载到本地 Document 目录。

(9) getImage 函数

加载图片的函数，判断本地是否有这个图片，如果有的话直接修改，否则调用 downloadImage 函数进行下载并设定。

(10) tappedMe 函数

表示图片被点按时发生的事件。

三、连接说明

1. ViewController 类

ViewController 继承了 UIViewController 类，对应一进入 App 时的场景，这个类控制主场景。

这个场景加载时会自动判断今日的壁纸是否下载到本地，如果下载到本地，则可以直接加载，否则下载后再加载。

(1) 输出口

①UIImageView!类变量 Image 连接到界面中的 UIImageView，用于加载背景图片

(2) 操作

①exitToHere 用于场景的回退

2. Gallery 类

Gallery 类继承了三个类，分别是：UIViewController, UITableViewDataSource, UITableViewDelegate。

Gallery 类对应点击 Gallery 所进入的场景，这个场景包含了一个列表式的浏览器即一个 UITableView 类的控件，App 会根据本地已有图片生成列表，并且按照月份划分同时按照时间倒序排列，为用户提供了较好的使用体验。

(1) 输出口

无

(2) 操作

无

3. ListView 类

ListView 继承了三个类：UIImagePickerControllerDelegate, UIViewController, UINavigationControllerDelegate。

ListView 负责 List 场景中的内容，主要包括通过一个 Stepper 对于当前图片的改变。

(1) 输出口

①UIImageView!类变量 Image 连接到界面中的 UIImageView，用于连接当前显示的图片

②UILabel!类型变量 currentDate，用于连接界面上表示日期的 Label

③UIStepper!类型变量 stepper，用于连接界面上的计数器

(2) 操作

①stepperChanged 操作。发生在 stepper 的 value changed 的事件，当 stepper 的值改变，对应改变类中的 current 变量，同时更新图片。

②saveButt 操作。发生在按钮保存图片点按时，可以将当前的 UIImageView 中的图片保存到本地，同时提示用户保存成功。

四、场景切换说明

1. ViewController 类

(1) 模态切换

①从 ViewController 可以通过点击 Gallery 按钮切换到 Gallery 的场景，采用模态切换，Presentation 类型为 Over Current Context，Transition 类型为 Cross Disslove。

②从 ViewController 可以通过点击 List 按钮切换到 ListView 的场景，采用模态切换，Presentation 类型为 Over Current Context，Transition 类型为 Cross Disslove。

(2) 回退

①exitToHere 用于其他场景的回退到当前场景。

2. Gallery 类

(1) present 弹出 ListView

通过点击 TableViewCell 的 Disclosure Indicator 弹出 ListView 界面，并且 ListView 中显示的图片即为当前选中的图片。

(2) 回退

①主页按钮连接到 ViewController 类中的 exitToHere 的 Exit，点击后可退回主界面。

3. ListView 类

(1) 模态切换

无

(2) 回退

①主页按钮连接到 ViewController 类中的 exitToHere 的 Exit，点击后可退回主界面。

五、应用逻辑说明

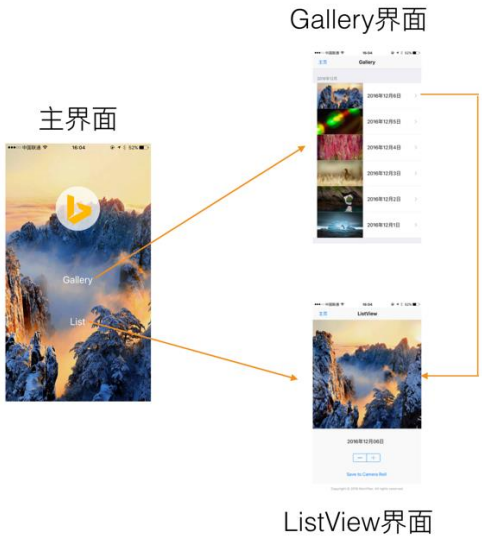

图 2

1. 交互逻辑

如图 2 所示，当用户进入 App 后呈现的是主界面，此时的背景是当天的必应主页壁纸，用户有两种交互选择。

①查看历史列表，即进入 Gallery 场景

②查看今日图片的完整样式，即进入 ListView 场景

当用户进入①时，即可以列表的形式呈现出所有已加载过的图片，当点击任一图片时，即可进入 ListView 场景查看这个图片的更详细的信息。

当用户进入②时，即可以两种不同的样式查看今日的图片，分别对应于 UIImageView 中的 scaleAspectFit 和 scaleAspectFill 属性。

此外，用户在 ListView 场景中，可以一键将照片储存到图库，大大提高了用户设置壁纸的效率。。

2. 图片加载逻辑

程序会先行判断该图片是否已经存储在本地，如果在本地即可直接读取，否则从网络获取。这样设计可以提高加载效率，同时减少用户的流量消耗。

3. Gallery 中列表生成逻辑

在 ViewDidLoad 方法中，程序会先判断有哪些图片可以供选择，然后生成这样的数组信息，存储在 months 和 dic 数组中，这两个数组前者表示的是月份，后者表示的是这个月份对应的元素的 id 信息。

TableView 会先得到栏目的数量即月份的数量，然后得到每个栏目中的条目数量即每个月份所对应的 id 数量，再得到每个栏目的标题即月份信息，最后得到每个条目的图片和文字信息。

4. ListView 中日期遍历的逻辑

在 ViewDidLoad 中，如果是从主页跳转过来的，则会自动将当前日期设置为今天所对应的年月日，否则则设置为从 Gallery 中传过来的值的信息。日期是 Date 类型，可以调用 addTimeInterval 方法对于 current 进行修改，由于 addTimeInterval 方法传递的是秒数，所以只需要使其值加减 24*60*60 秒即可进行日期的前后加减。

六、相关技术说明

1. 图片的点按手势

通过代码的方式开启了 UIImageView 的交互，同时本人自己增加了 TapGesture 的交互方式，使程序更加人性化

2. ListView

ListView 对应的内容是书中第 14 章的内容，本人自学了相关章节并将这个控件应用到了程序中。

3. 离线存取的逻辑设计

在最初的版本中，我统一让程序现场读取服务器中的数据。但是考虑到加载需要时间，等待往往给用户不好的体验，同时当数据量大的时候还会加重服务器的负载，此外，从我个人使用 app 而言，我还是很关注 app 对于手机流量的消耗情况的。所以，因为有这个需求，我同样自学了相关章节，实现了本地存取数据，当第一次从服务器读时立即下载，之后除非访问之前未加载过的照片，否则不会发生与服务器的数据交换。

4. 交互逻辑

在 app 中往往是两个场景来回切换，或者多个场景有顺序的进行切换，形成栈式交互顺序，很少见三个场景来回切换的，而该 App 就采用的这样的一种交互逻辑，通过 present 方法来实现，虽然从第三个场景直接切换回第一个场景时，第二个场景会昙花一现，不过这并不影响日常使用。

5. 图片自动更新

这个功能很好实现，只需要每天获取当天的日期即可。但是对于定位于每天打开一次的 app 来讲，高质量的图片会每天给人带来耳目一新的感觉，这个是我个人比较喜欢的。

七、总结

1. 设计理念

这款 App 的定位在于，用户每天打开一次，如果喜欢当天的壁纸，则可以仔细浏览这张图片的全貌，如果满意的话可以直接保存到相册里，然后设置为自己的壁纸。此外，如果用户误删了之前保存过的喜欢的图片，可以去列表中根据小图浏览，定位到那张壁纸，从而保存到本地。

这款 App 专注的解决了上面的两个需求，我相信拥有这个需求的人群相比于其他同学设计的自娱自乐的 App 的需求人群要大得多。同时有以下几个原则也是我做的时候尽量克制的。

第一，保持界面绝对的干净，没有用的按钮一律不要。对于大部分的 App，用户只会在 20%的功能上花费 80%的时间，而本人设计的这款 App 力求用户在 80%的功能上花费 80%的时间。本人一直认为，App 的真正使用效率才是衡量 App 优劣的关键。

第二，减少用户的使用时间。这点和第一点类似，但是本人希望这个 App 是带给用户以便利的，每天可能只用花 1 分钟打开 App，流览图片，然后保存到本地，设置背景图片，这样每天打开手机都有一种耳目一新的感觉。这一分钟可以出现在等车时、等电梯时，简单便捷。

第三，将控件放在合适的地方。当我看了很多同学设计的 App，发现很多人都是为了凑控件而用控件，设计的完全不合理，可以说为了作业要求而有意设置，但是本人一直认为，一个 App 如果功能并不够实用，便利程度不够高，则并不能称为一个 App。对于我这款 App 来讲，选择日期为什么要用一个 Stepper，而不用一个日期选择器，即便这样做是完全可以的，但是对于图片的预览，显然是要一张一张的浏览，而并不是一下跳跃好几个月去浏览，所以考虑再三，还是坚守了这个原则，使用了 Stepper 控件，如果需要指定日期，那么 Gallery 场景的 ListView 控件给用户的体验会远比日期选择器强。

2. 问题

大部分问题都可以通过 Google 和 StackOverflow 解决，但是有两个事情仍然没有解决。

①界面切换问题

这是一个交互性的问题。从 ViewConrtoller 切换到 Gallery，然后再从 Gallery 切换到 ListView 中，然后想从 ListView 一步切换回主界面，中间会出现 Gallery，这个问题没有在网上找到比较好的解决方法。

②无法在 app 内设置壁纸

这是一个功能性的问题。本人之前使用其他 App 的时候也没有见到能直接在 App 中设置壁纸的，此外根本查不到相关的信息。所以本人怀疑苹果并没有提供这样的一个接口供开发者使用。

3. 代码说明

对于代码的解释在源程序中有较为清楚的解释，限于文章篇幅，不刻意粘贴代码于此报告中。敬请读者访问 GitHub 项目主页阅读源码。

4. 感想

关于必应主页的一系列事情我早在暑假就有着手，爬取网站图片后一直没有地方使用。当前几节课加载了网络获取的图片的时候，我就觉得做一个关于必应壁纸的 App 是可行的。关于这种每日 App 也受到了应用商店中的很多 App 启发，如“每日一言”等，这些 App 大都功能简单，但是非常精美，可以让生活更加美好同时又不占用过多时间。我一直也相信相对于电脑端复杂的功能而言，移动端就应当做些简单有趣的事情，但是由于移动端的便携与较低的经济成本，这个势头似乎恰恰相反。

这是第一次部分出于个人使用需求而做的 App，这也是本人第一次实现本地与服务器端内容交互的项目。

非常感谢龙腾老师教授我这门课程并且此项目的指导。

非常感谢悉心拍摄挑选必应主页壁纸的艺术家们。