

1.1 创建蓦然项目

运行Xcode,在File菜单中选择New→Project...

Xcode显示新的工作窗口，选择位于下拉窗口左侧iOS栏下的Application,右侧有若干应用模板可供选择，选择Single View Application(单视图应用)。

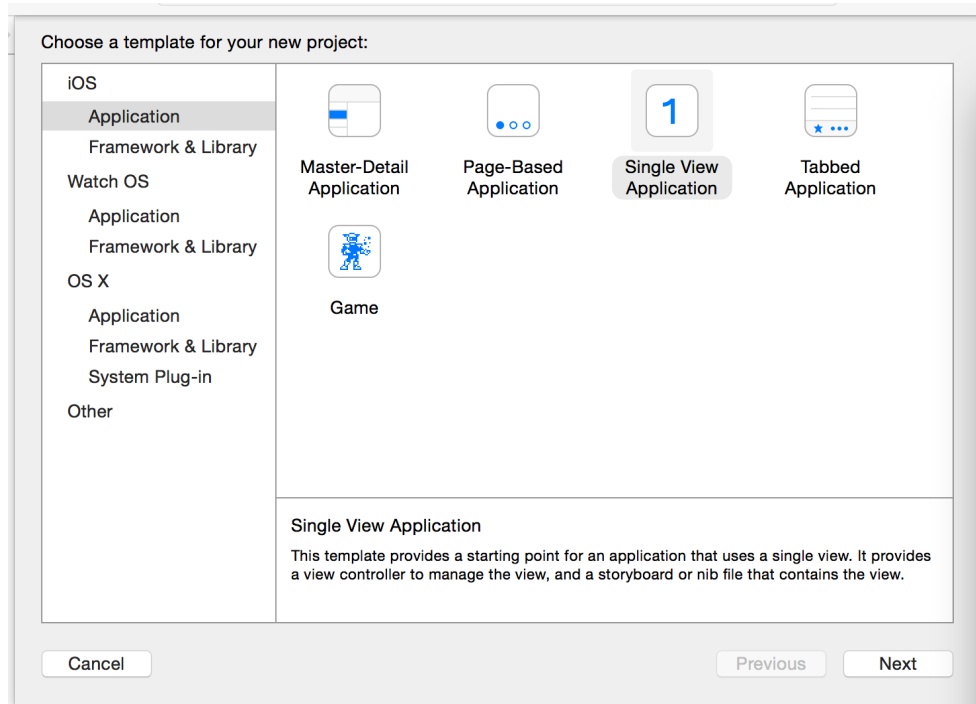


图 1-1 设置新项目

点击Next按钮，在新出现的界面中，将“蓦然”填入Product Name文本框(见图1-2)，Organization Name和Organization Identifier可以填入 com.GeekBand。也可以填入自己的公司名称。

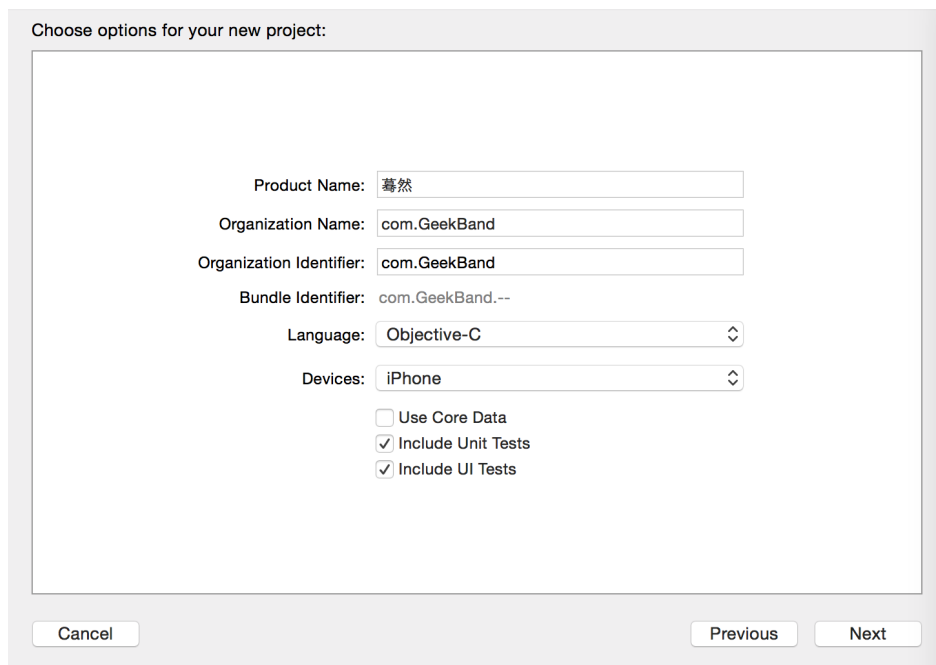


图 1-2 命名新项目

Language选择Objective-C，Devices选择iPhone。单击Next按钮，显示最后一个界面，保存项目。单击Create， 蓦然项目就创建好了。

1.2搭建基本框架

在项目导航面板中找到系统自动生成的ViewController.h和ViewController.m文件和Main.storyboard文件并删除。

在项目导航面板中的蓦然文件夹上，右键选择Show in Finder，添加Models目录、Views目录、Controllers目录、Resources目录、APP目录、Supporting Files目录，如图(1-3)

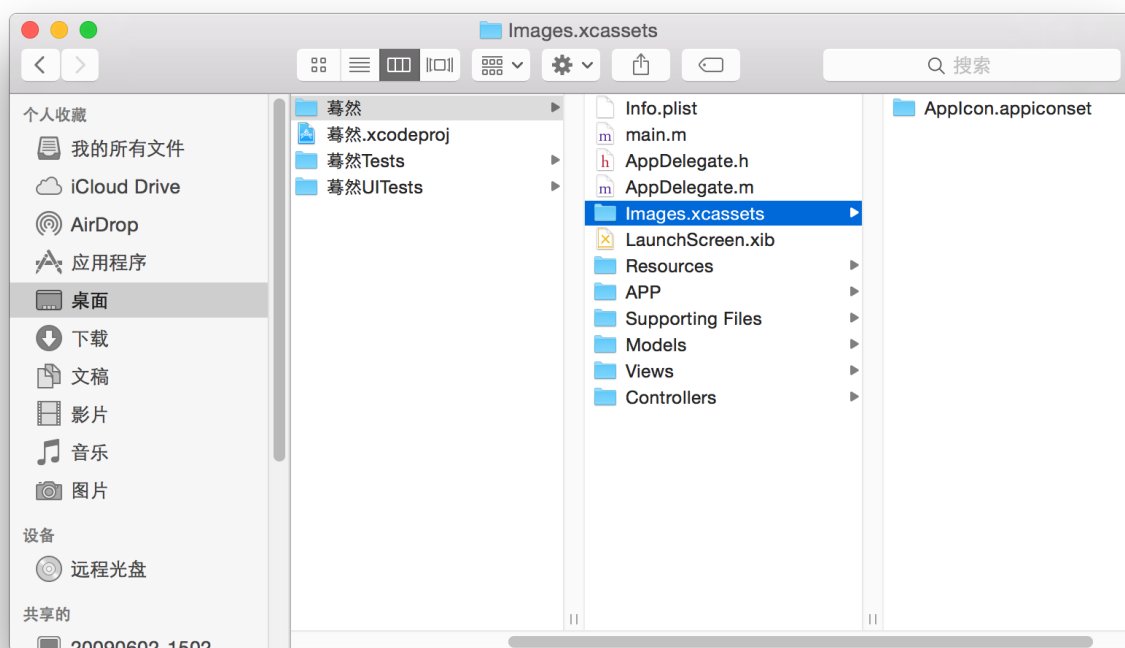


图 1-3创建新目录

将info.plist、main.m放入Supporting Files目录,AppDelegate.h和AppDelegate.m放入APP目录， Images.xcassets和LaunchScreen.xib放入Resources目录后， 回到Xcode中项目导航面板， 在蓦然文件上右键选择Add Files to “蓦然”...

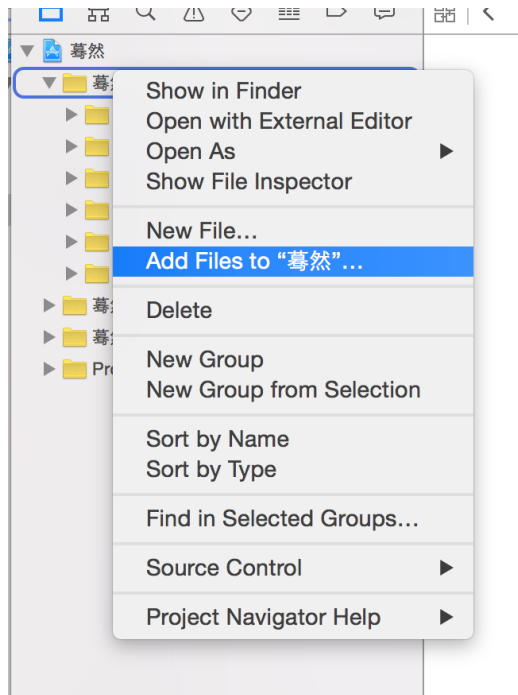


图 1-4导入新目录

选中之前加入的目录，点击**Add**键，完成架构目录的导入。如果正确导入，应如下图所示

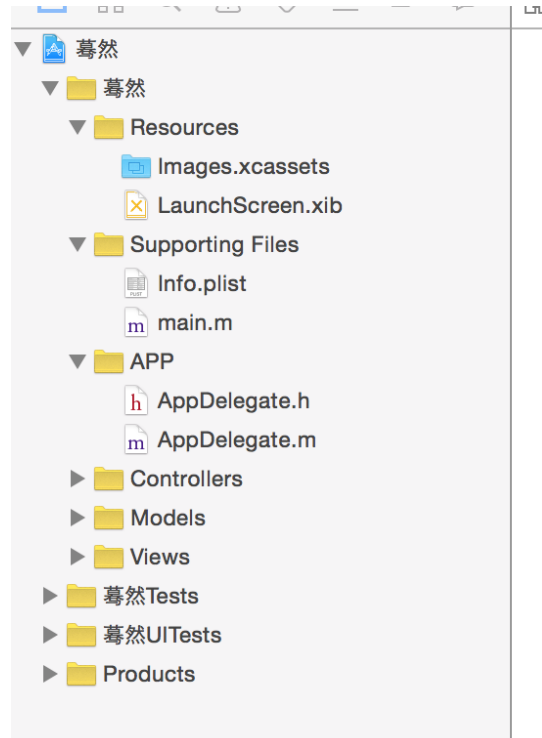


图 1-5完成后目录结构

2.1 准备创建登陆模块

在创建界面之前，先开始准备工作，下载GeekBand的资源包，将资源包下的界面图片文件导入Main.xcassets。

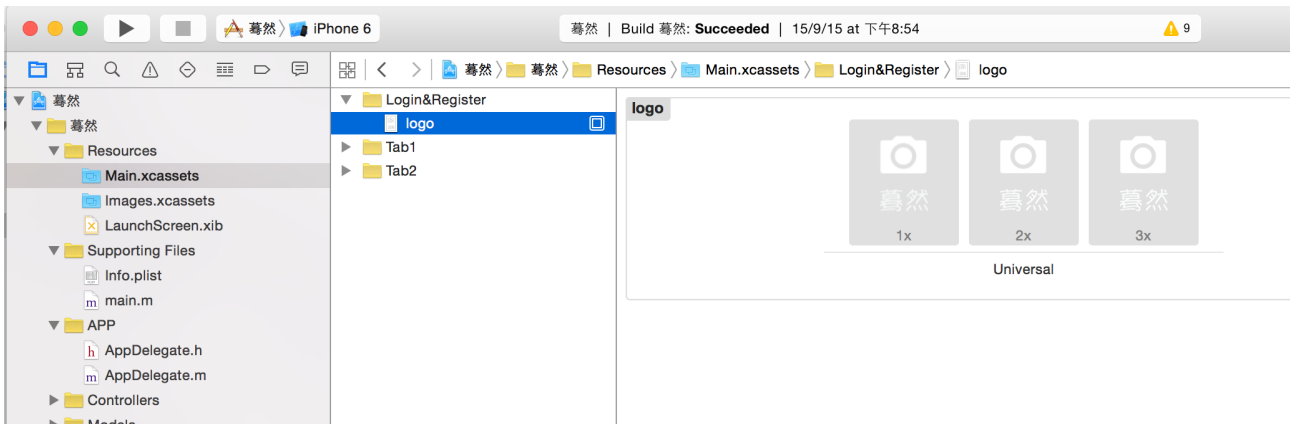


图 2-1 导入界面资源包

在项目导航面板中找到Views目录中，在该目录下，新建“LoginAndRegister”目录，创建New File，在左侧iOS目录下的User Interface，在右侧中选择Storyboard，点击Next按钮。

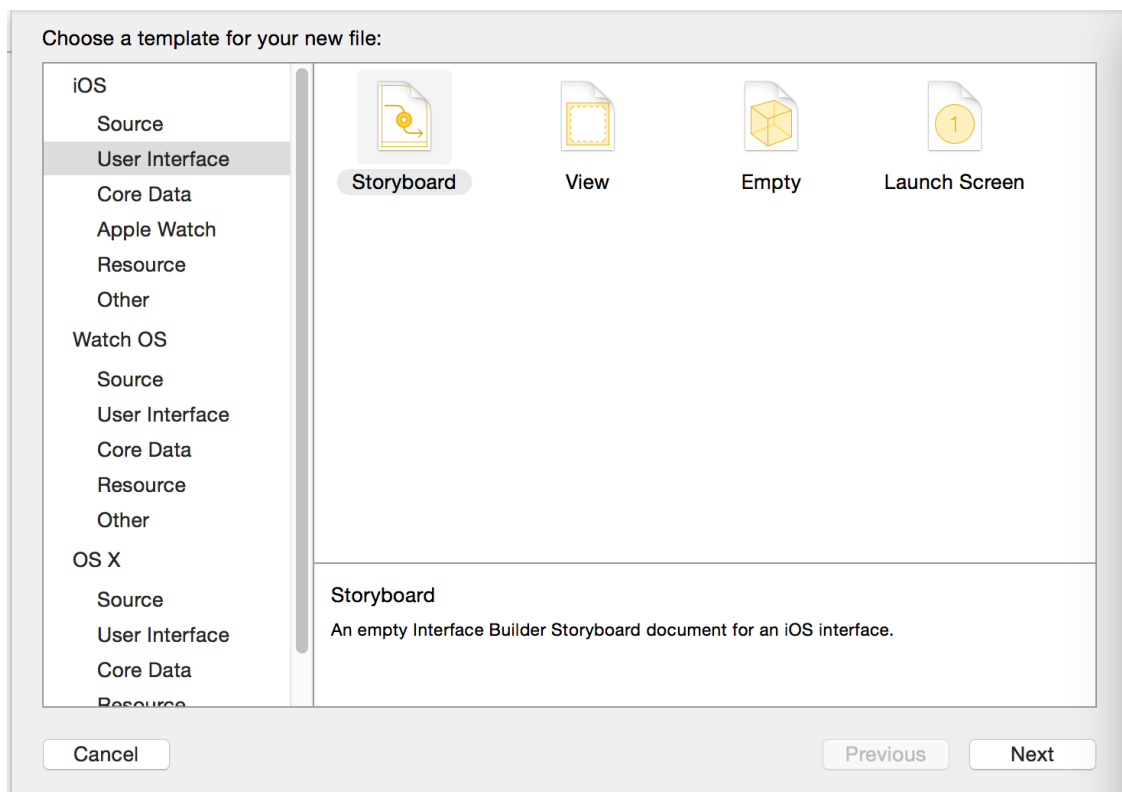


图 2-2创建Storyboard

在选择保存位置界面中，**Save As**文本框中输入“LoginAndRegister”，点击**Create**。完成后如图

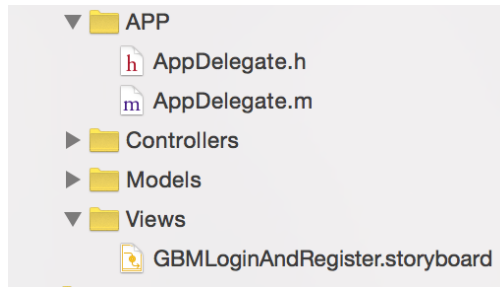


图 2-3完成创建图

在构建Storyboard文件前，先将我们的应用和原来默认的Main.storyboard取消关联。选中项目导航面板顶部的蓦然项目文件，在TARGETS中选择蓦然项目文件，再点击**General**标签，找到标题为Main Interface的文本框，将默认的Main置为空。

通过新创建的Storyboard文件，不用编写任何代码，打开工具区域，拖拽一个**ViewController**对象至画板。

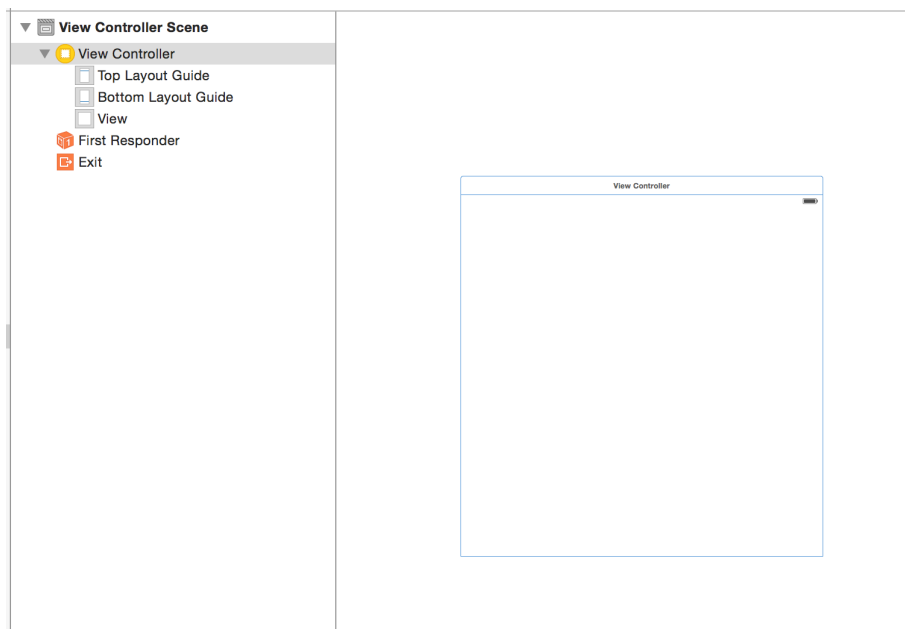


图 2-3完成Storyboard中的ViewController对象

在项目导航栏目中ViewController目录下New Group 相关目录“LoginAndRegister”，在该目录下New File，选择位于下拉窗口左侧IOS栏下的Source,选择Cocoa Touch Class，点击Next。

在Subclass of下拉列表中选择UIViewController，Class的名字选择“GBMLoginViewController”，Language选择OC。

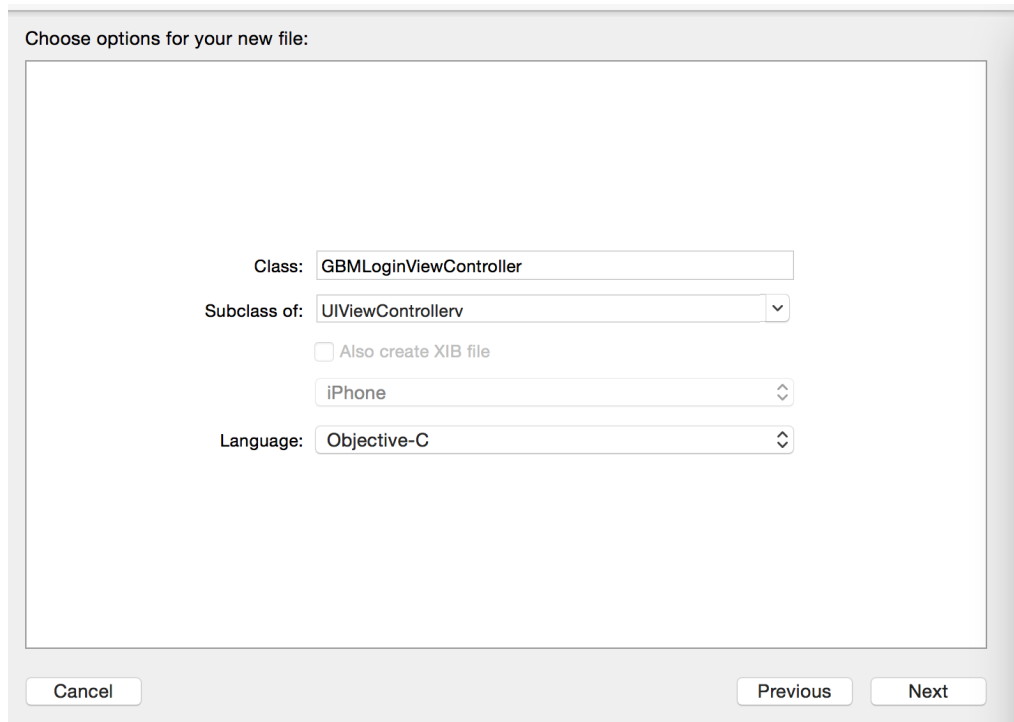


图 2-3创建ViewController对象

完成创建后，目录应该如图(2-5)所示。

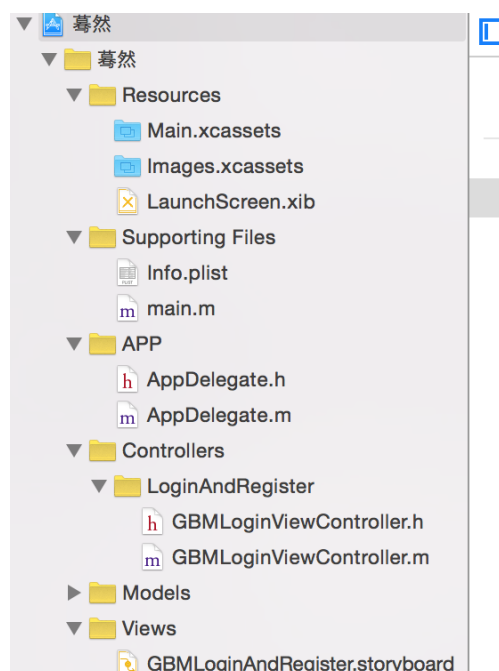


图 2-4查看现时目录结构

现在打开，LoginAndRegister.storyboard文件，点击UIViewController对象，然后打开标识检视面板，找到标题为Class的文本框，将其设置为GBMLoginViewController。再将Storyboard ID设置为LoginStoryboard，并将Use Storyboard ID打钩。

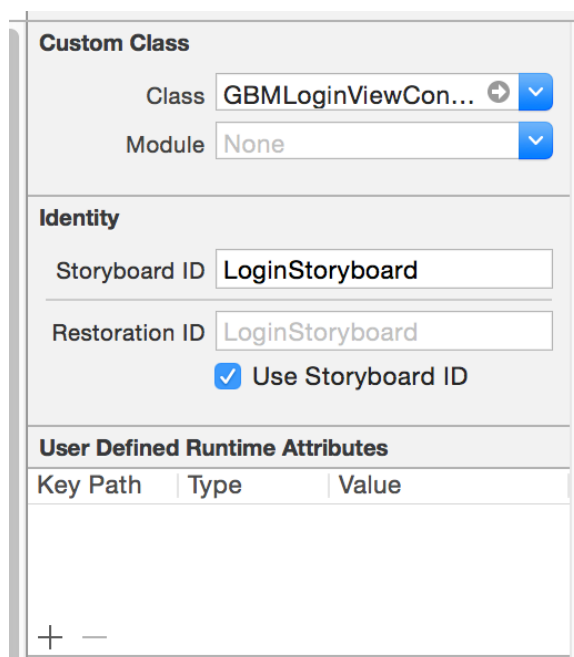


图 2-5 设置Storyboard的ID

至此，完成Storyboard与Class之间的关联，下面开始完成组装界面的工作。在最后，在TAGETS中选择蓦然项目文件，再点击General标签，找到标题为Main Interface的文本框，填入LoginAndRegister。

2.2 创建视图对象

从对象库面板(位于Xcode右下方)拖拽一个UIViewController对象至画布。在右侧工具栏上,选择Attribute inspector页签,在Simulated Metrics选择尺寸为iPhone 4-inch。

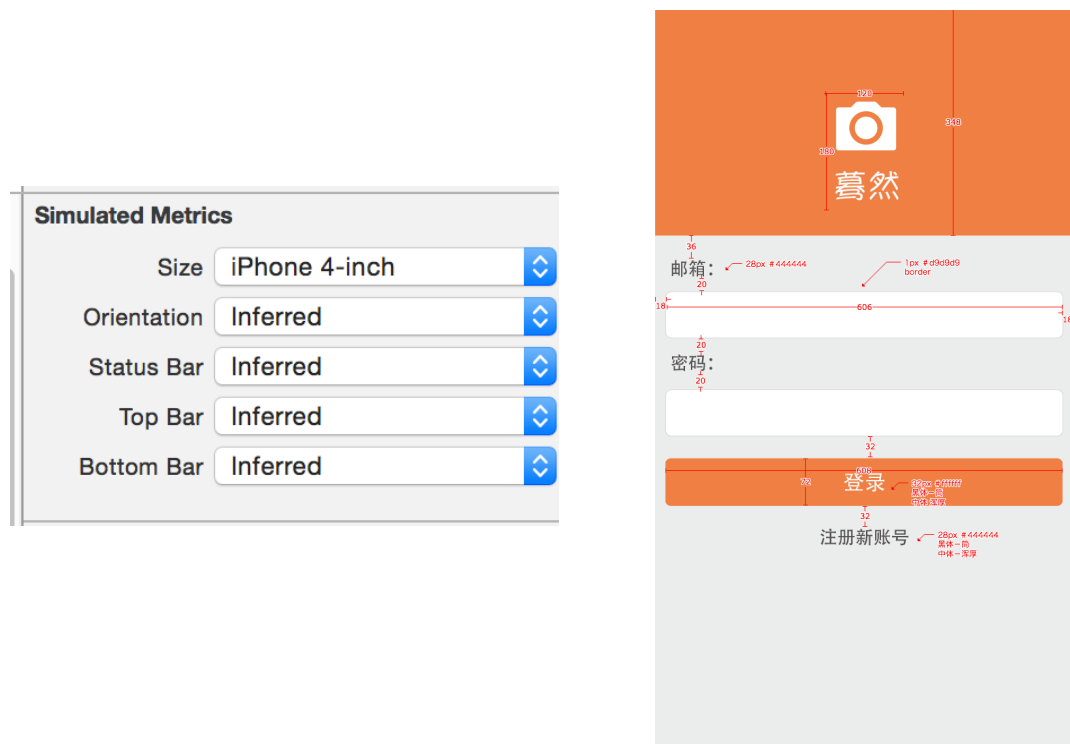
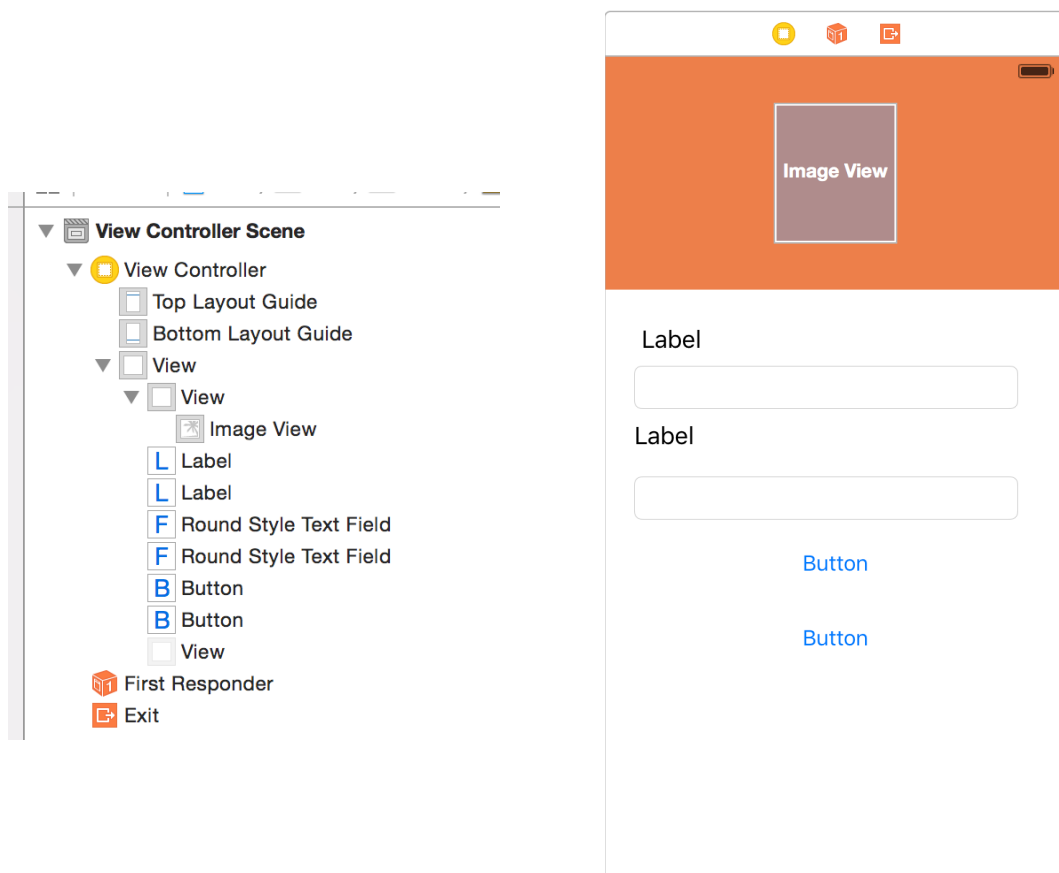
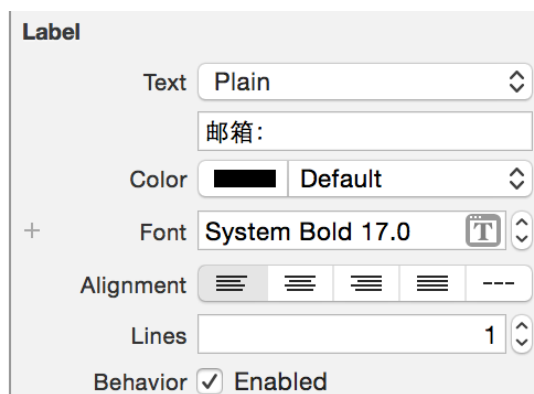


图 2-6 设置viewController尺寸和设计图

然后拖拽一个UIImageView对象、两个UILabel对象、两个UIButton对象、两个UITextField对象和一个子UIView对象,置入UIView对象,并设置其大小和位置后,如图



其中统一设置字体为**System Bold 17.0**

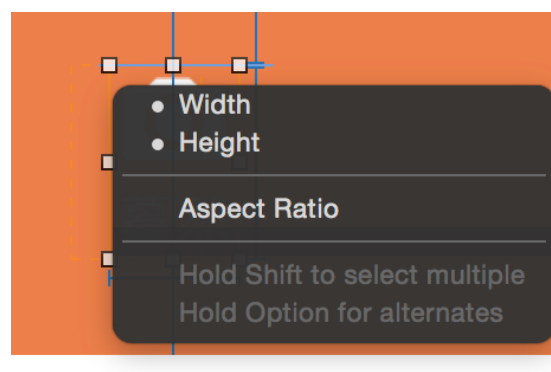
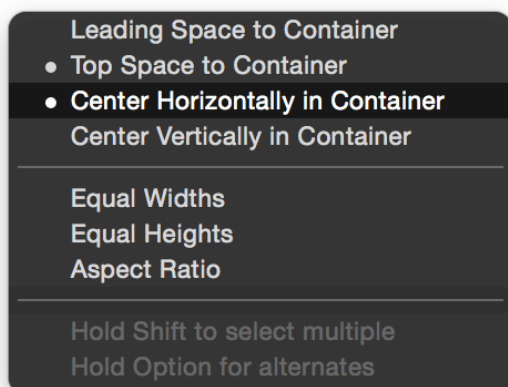



双击Button按钮，修改其标题为“登陆”和“注册新账号”，双击Label对象，修改其text属性为“邮箱”和“密码”，将视图上半部的子UIView和登陆按钮的RGB背景色设置为(230, 106, 58),将UImage对象的image属性设置为“logo”，完成后应如下：



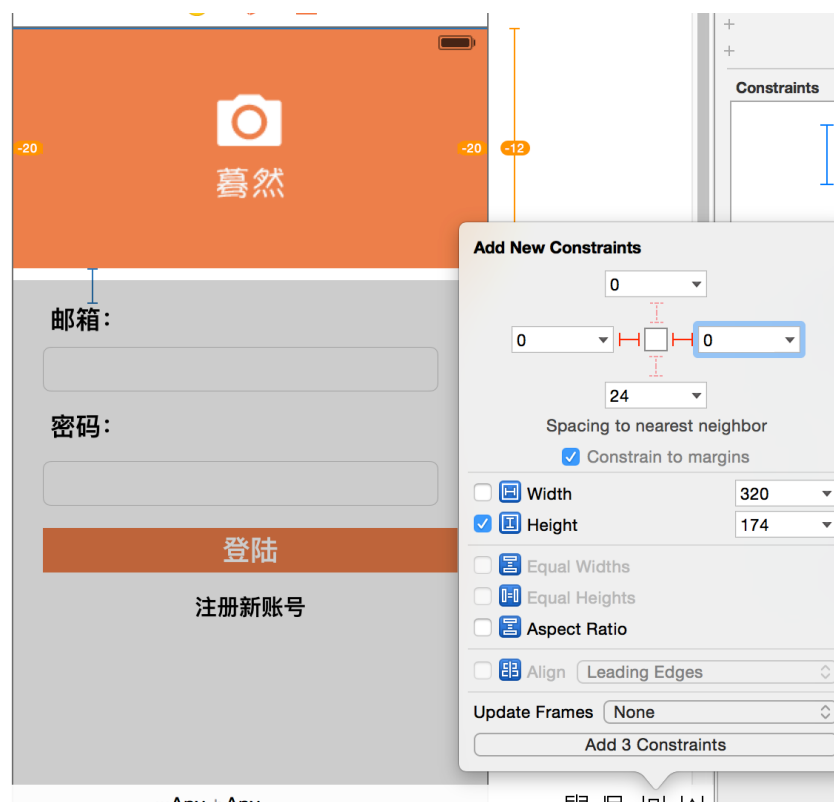
2.3 自动化布局

点击logo图标，按住Control,左斜上角度拖拽至背景子UIView，显示菜单，选择Top Space to Container后，再次同样操作，选择Center Horizontally in Container。然后再次点击logo图标，按住Control，左斜上角度拖拽至自身，选择Width和Height，完成对logo的约束，如图所示：

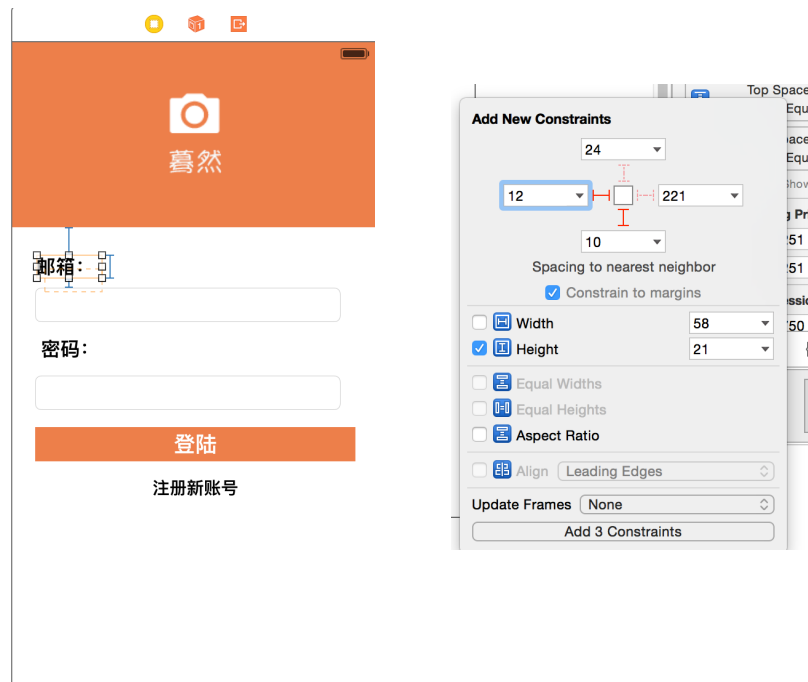


点击子View，再  点击图标(左边第三个)，显示Pin菜单，Pin菜单显示了子UIView对象的大小和位置，可以在菜单为其对象添加需要的约束。这里设置左边，上边，右边与最相邻视图的距离，由于这三个方向没有兄弟视图，所以这三个方向的相邻视图为GBMLoginViewController的view，底部距离设为24，高度设定为174。

点击输入框和小矩形之间的橘红色虚线，虚线会变成实线。这样就可以将输入框中的值添加为视图的约束。



下面为“邮箱”标签和密码标签添加约束，如果不考虑语言、字体，“邮箱”标签应该始终处于屏幕左上角而且大小始终保持不变，在画布中选择“邮箱”标签，然后点击Pin菜单。

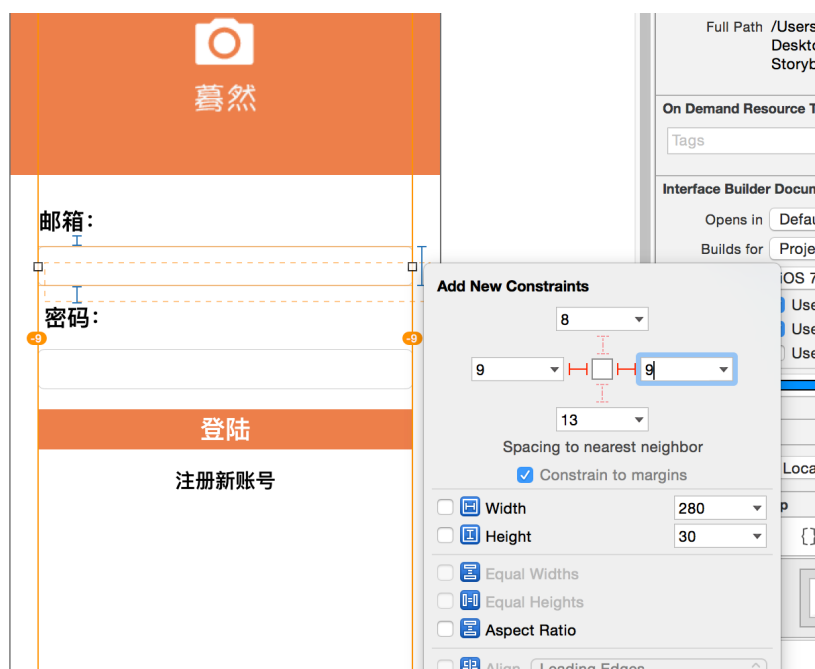


请注意，约束的值是根据视图在画布上的位置计算出来的，读者的“邮箱”标签位置可能与图中不一致，所以约束的值也不同。如果读者的Pin菜单中的输入框的值，不需要将其修改成图中的值。否则“邮箱”标签的约束和画布上的位置不一样，出现报错。

现在文本框的约束出现了问题。画布中表示文本框约束的直线在正常情况下是蓝色的，但是现在变成了橘红色的，这表示文本框缺少约束。可以在Dock中点击View Controller Scene的红色图标进入约束问题列表。




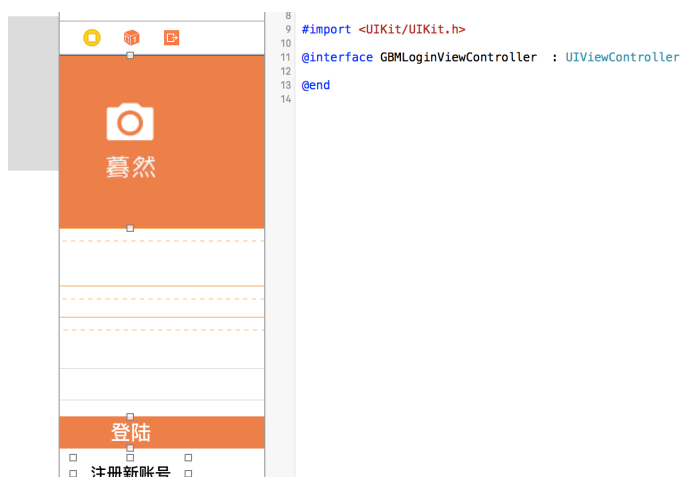
文本框的约束问题属于Missing Constraints(缺少约束)。根据问题描述可以知道，目前文本框缺少X轴的约束。解决方法是，在Pin菜单，添加左右约束。



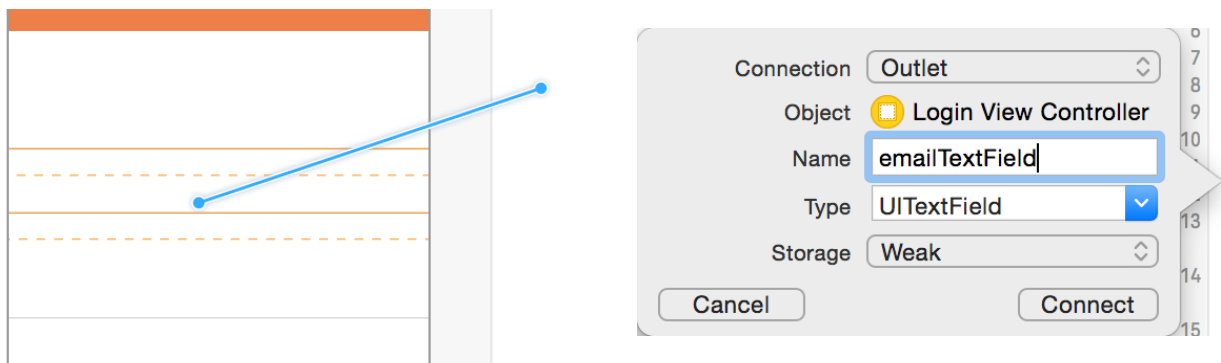
下面读者根据上面的范例，将剩下的视图添加类似的约束。完成后，构建并运行，应如与在Storyboard中一致。

2.4 添加响应事件

打开GBMLoginAndRegister.storyboard文件，打开位于Xcode右上角的  图标，在界面右边会弹出关联的Class的文件代码，如图所示。

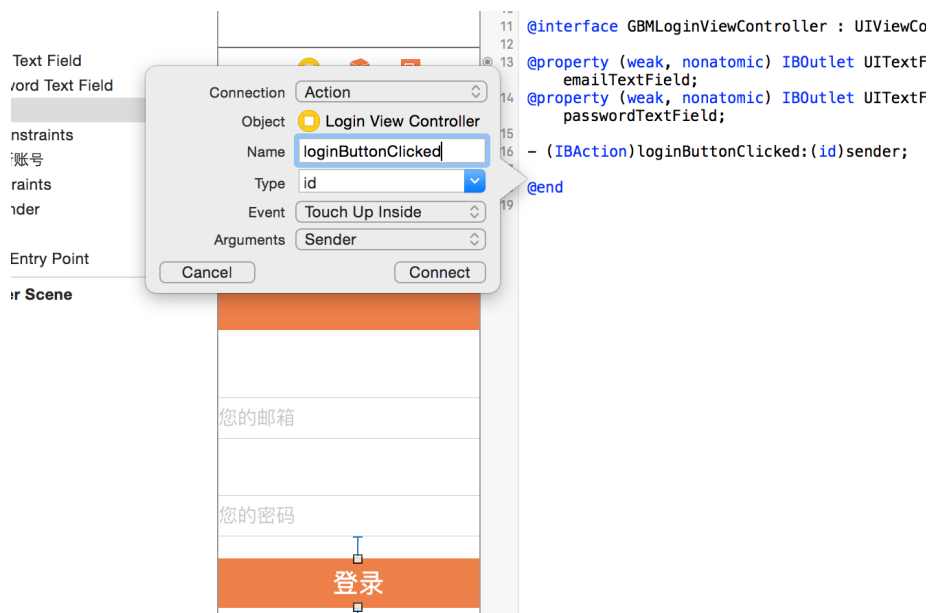


下面将为textField设置插座变量，点击邮箱的textField，按住Control，拖拽到弹出的GBMLoginViewController中，在@interface 和@end的位置之间。松开后，在关联菜单中的Name输入框中，输入“emailTextField”。点击Connect，完成插座变量设置。



请读者使用同样的方式为密码的textField创建一个类似的关联。注意，这里蓝线的起点是“拥有插座变量的对象”，终点是“插座变量需要指向的对象”。

然后再为按钮设置动作方法，先要设置对象目标，可以点击登录按钮后按住Control键，拖拽该对象如插座变量一样到GBMLoginViewController，命名为“loginButtonClicked”。



需要注意的关联动作方法时,Connection下拉菜单中应选择Action。而“注册新账户”按钮,这里我们暂时不为其添加类似的关联动作,留置注册模块。

完成界面控件的关联后,可以开始实现动作方法,先为登录按钮,添加验证功能,防止邮箱或密码的输入框为空。在GBMLoginViewController中的loginButtonClicked添加以下代码。

```
40 - (IBAction)loginButtonClicked:(id)sender
41 {
42     NSString *userName = self.userNameTextField.text;
43     NSString *password = self.passwordTextField.text;
44
45     // 验证邮箱和密码是否都有输入内容,且检查邮箱格式是否正确
46     if (([userName length] == 0) ||
47         ([password length] == 0)) {
48         NSLog(@"失败");
49     } else {
50         NSLog(@"成功");
51     }
52 }
```

构建并运行,逻辑应能正确执行。然后来为UITextField设置键盘收回的处理。下面为UITextField对象所位于的视图控制器设置它的delegate,并实现textFieldShouldReturn: 委托方法,点击Return,就会收回键盘。

打开GBMLoginViewController.h,如下设置。

```
1 @interface GBMLoginViewController : UIViewController<UITextFieldDelegate>
2
3 @property (weak, nonatomic) IBOutlet UITextField *emailTextField;
4 @property (weak, nonatomic) IBOutlet UITextField *passwordTextField;
5
```

再回到GBMLoginViewController.m文件,进入如下设置。

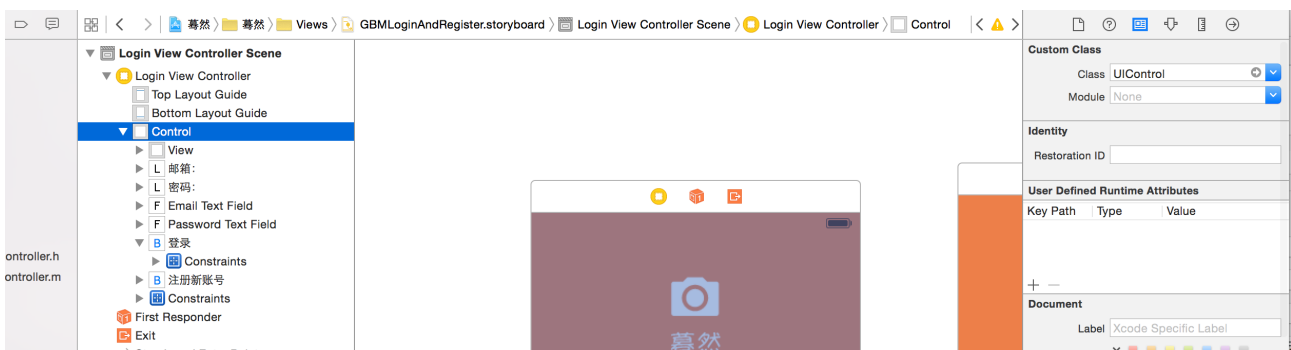
```
- (void)viewDidLoad
{
    [super viewDidLoad];
    self.emailTextField.delegate = self;
    self.passwordTextField.delegate = self;
}
```

```

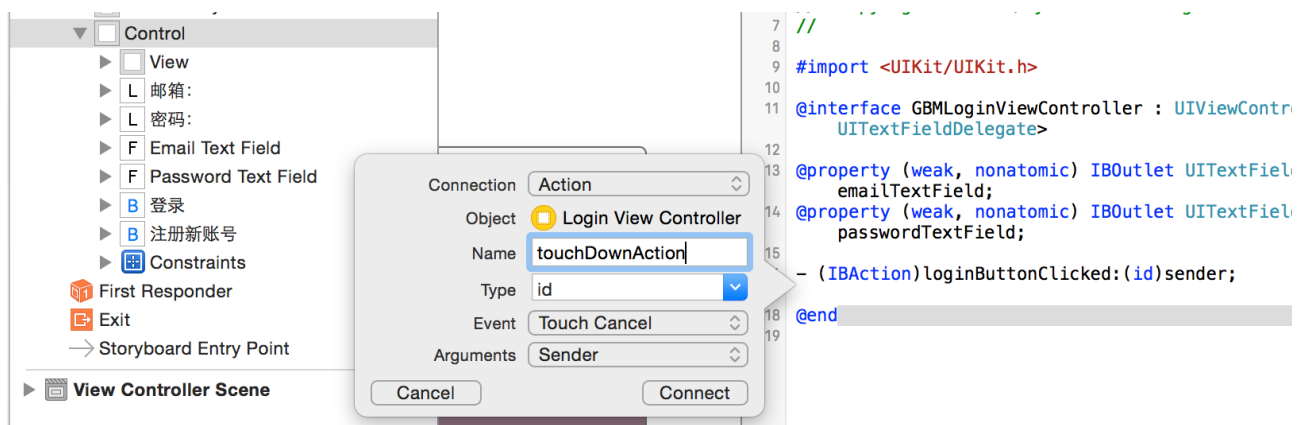
77 // 通过代理来让键盘上的return键实现关闭键盘
78 - (BOOL)textFieldShouldReturn:(UITextField *)textField
79 {
80     [textField resignFirstResponder];
81     return YES;
82 }

```

构建并运行，应能在textField中点击Return，键盘可以成功收起，不过这显然是不够的，正确地交互逻辑应该是在打开键盘后，点击屏幕其它位置能收起键盘。打开GBMLoginAndRegister.storyboard。将UIView设置为UIControl。



点击Control，按住Control，将其关联到GBMLoginViewController，其中Event设为Touch Down，名字设为touchDownAction。



回到GBMLoginViewController.m文件，进入如下设置。


```

70
71 - (IBAction)touchDownAction:(id)sender
72 {
73     [self.emailTextField resignFirstResponder];
74     [self.passwordTextField resignFirstResponder];
75 }
76
77 // 通过代理来让键盘上放... 键盘相关的操作

```

再次构建并运行，程序打开键盘后点击其它位置，应能自如收回键盘。

回到登陆按钮，下面将为应用的登陆按钮添加更为详细的逻辑代码。我们为验证失败设置方法“showErrorMessage”,为验证成功设置“loginHandle”。

在GBMLoginViewController.m文件，添加如下代码。

```

73
74
75 // 创建一个弹出UIAlertView的方法，用来提示用户
76 - (void)showErrorMessage:(NSString *)msg
77 {
78     UIAlertView *alert = [[UIAlertView alloc] initWithTitle:nil
79                                     message:msg
80                                     delegate:nil
81                                     cancelButtonTitle:@"确定"
82                                     otherButtonTitles:nil];
83     [alert show];
84 }
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

并且将之前写的loginButtonClicked修改为：

```

9
10 - (IBAction)loginButtonClicked:(id)sender
11 {
12     NSString *userName = self.userNameTextField.text;
13     NSString *password = self.passwordTextField.text;
14
15     // 验证邮箱和密码是否都有输入内容，且检查邮箱格式是否正确
16     if (([userName length] == 0) ||
17         ([password length] == 0)) {
18         [self showErrorMessage:@"邮箱和密码不能为空"];
19     } else {
20         [self loginHandle];
21     }
22 }
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

构建并运行，程序可以在登录成功应能输出登录信息，在验证失败能弹出警告信息。

2.5网络编程

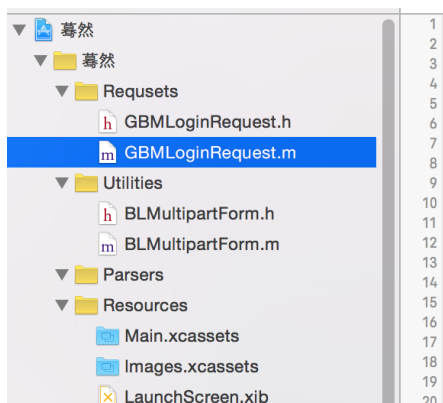
前文已经完成了登陆的验证，现在缺少的是真正与服务端进行连接。这里选择使用段松老师在网络编程课所讲授的网络编程框架。

在项目导航面板中的蓦然文件夹上，再新创建三个目录，分别为“Parsers”，“Utilities”，“Requests”。

在Requests目录下创建一个名为GBMLoginRequest的NSObject子类。其对象负责向服务端发送请求，然后在GBMLoginRequest.h添加如下代码。

```
9  #import <Foundation/Foundation.h>
10
11 @interface GBMLoginRequest : NSObject
12
13 @property (nonatomic, strong) NSURLConnection *urlConnection;
14 @property (nonatomic, strong) NSMutableData *receivedData;
15
16 - (void)sendLoginRequestWithUserName:(NSString *)userName
17                                     password:(NSString *)password
18                                     gbid:(NSString *)gbid;
19
20
21
22 @end
23
```

由于我们使用的是POST请求，我们在Utilities目录下导入段松老师的BLMultipartForm文件，可从资源包找到。



于是我们在GBMLoginRequest.m 中#import “BLMultipartForm.h”。
然后在GBMLoginRequest.m文件中，实现sendLoginRequest方法。代码如下：

```
[self.urlConnection cancel];

NSString *urlString = @"http://moran.chinacloudapp.cn/moran/web/user/login";

// POST请求
NSString *encodeURIComponent = [urlString stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];

NSURL *url = [NSURL URLWithString:encodeURIComponent];
NSMutableURLRequest *request = [[NSMutableURLRequest alloc] initWithURL:url];
request.HTTPMethod = @"POST";
request.timeoutInterval = 60;
request.cachePolicy = NSURLRequestReloadIgnoringLocalAndRemoteCacheData; // 忽略本地和远程的缓存

NSData *data1 = [request HTTPBody];

BLMultipartForm *form = [[BLMultipartForm alloc] init];
[form addValue:email forKey:@"email"];
[form addValue:password forKey:@"password"];
[form addValue:gbid forKey:@"gbid"];
request.HTTPBody = [form httpBody];
[request setValue:form.contentType forKey:@"Content-Type"];

self.urlConnection = [[NSURLConnection alloc] initWithRequest:request
                                                         delegate:self
                                                         startImmediately:YES];
```

再为其NSURLConnection设置delegate方法。先在GBMLoginRequest.h中添加如下

```
9  #import <Foundation/Foundation.h>
10
11 @interface GBMLoginRequest : NSObject<NSURLConnectionDataDelegate>
12
13
14 @property (nonatomic, strong) NSURLConnection *urlConnection;
15 @property (nonatomic, strong) NSMutableData *receivedData;
```

然后回到GBMLoginRequest.m中实现delegate方法。

```
0
1 #pragma mark - 网络请求代理方法
2
3 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
4 {
5     NSHTTPURLResponse *httpResponse = (NSHTTPURLResponse *)response;
6     if (httpResponse.statusCode == 200) {
7         self.receivedData = [NSMutableData data];
8     }
9 }
10
11 - (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
12 {
13     [self.receivedData appendData:data];
14 }
15
16 - (void)connectionDidFinishLoading:(NSURLConnection *)connection
17 {
18     NSString *string = [[NSString alloc] initWithData:self.receivedData encoding:NSUTF8StringEncoding];
19     NSLog(@"receive data string:%@", string);
20 }
21
22
23
24 - (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
25 {
26     NSLog(@"error = %@", error);
27 }
28
```

成功发送网络请求后，下一步我们需要来创建一个类用来解析json数据，另外一个Model类用来装载数据。首先在Models目录下再创建一个User目录，在该目录下创建一个名为“GBMUserModel”的NSObject子类。

在GBMUserModel.h添加用户模型的属性：

```
9  #import <Foundation/Foundation.h>
10
11  @interface GBMUserModel : NSObject
12
13  @property (nonatomic, copy) NSString *username;
14  @property (nonatomic, copy) NSString *email;
15  @property (nonatomic, copy) NSString *password;
16  @property (nonatomic, copy) NSString *loginReturnMessage;
17  @property (nonatomic, copy) NSString *registerReturnMessage;
18
19  @end
20
```

然后再在Requests目录下创建名为“GBMLoginRequestParser”的NSObject子类。在其GBMLoginRequestParser.h文件和GBMLoginRequestParser.m文件添加如下代码：

```
1
2  // 这个类是把JSON数据转换成model
3  @interface GBMLoginRequestParser : NSObject
4
5  - (GBMUserModel *)parseJson:(NSData *)data;
6
7  @end
8
```

```

#import "GBMLoginRequestParser.h"

@implementation GBMLoginRequestParser

- (GBMUserModel *)parseJson:(NSData *)data
{
    NSError *error = nil;
    id jsonDic = [NSJSONSerialization JSONObjectWithData:data
                                                         options:NSJSONReadingAllowFragments
                                                         error:&error];

    if (error) {
        NSLog(@"The parser is not work.");
    } else {
        if ([[jsonDic class] isKindOfClass:[NSDictionary class]]) {
            id returnMessage = [jsonDic valueForKey:@"message"];
            if ([[returnMessage class] isKindOfClass:[NSString class]]) {
                GBMUserModel *user = [[GBMUserModel alloc] init];
                user.loginReturnMessage = returnMessage;

                return user;
            }
        }
    }
    return nil;
}

@end

```

现在我们可以发送登陆请求，解析json数据，填充模型，最后要做的就是完善GBMLoginViewController的登陆功能。回到GBMLoginViewController.m文件，在文件开头#import “GBMUserModel.h”,#import “GBMLoginRequest.h”，在类扩展中添加属性

```

@interface GBMLoginViewController ()

@property (nonatomic, strong) GBMLoginRequest *loginRequest;

@end

```

并完善loginHandle方法：

```

2 // 核对用户的登录信息
3 - (void)loginHandle
4 {
5     NSString *email = self.emailTextField.text;
6     NSString *password = self.passwordTextField.text;
7     NSString *gbid = @"GeekBand-I150001";
8
9
10    self.loginRequest = [[GBMLoginRequest alloc] init];
11    [self.loginRequest sendLoginRequestWithEmail:email
12                                     password:password
13                                     gbid:gbid];
14
15    GBMUserModel *user = [[GBMUserModel alloc] init];
16    if ([user.loginReturnMessage isEqualToString:@"Login success"]) {
17        NSLog(@"登录成功");
18        [self showErrorMessage:@"登录成功了吗? "];
19    } else {
20        NSLog(@"服务器返回值: %@", user.loginReturnMessage);
21    }
22
23 }

```