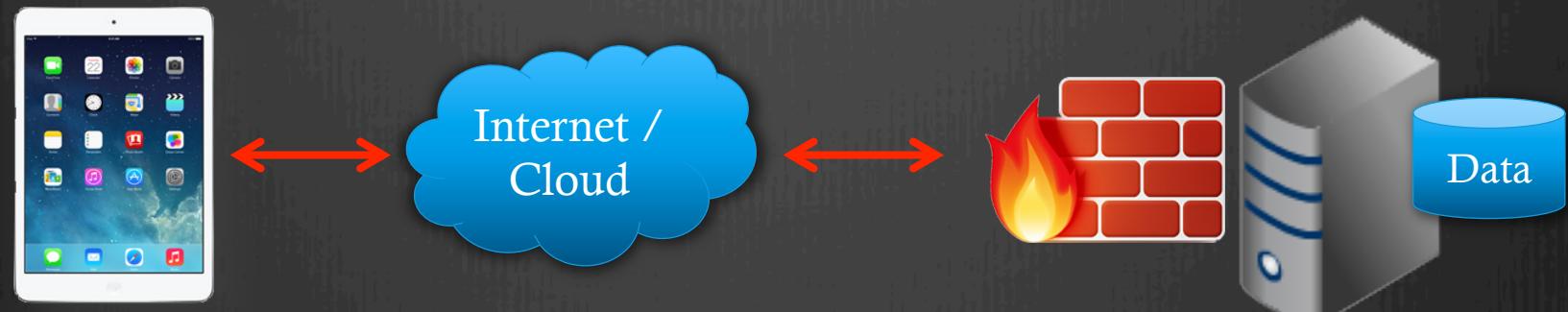


Chapter 14

iOS RESTful Client + JSON

Client + Server + Web



- HTTP Based
- Web Services / RESTful
- XML / JSON

RESTful

- ◉ RESTful ย่อมาจาก Representational State Transfer (หรือ REST)
- ◉ มาตรฐานของ REST ถูกกำหนดโดย W3C เพื่อเป็นทางเลือกในการ implement solution แบบ web service ด้วยเทคโนโลยีของ web ที่มีอยู่แล้ว
- ◉ เน้นความเรียบง่าย ไม่มีกฎเกณฑ์มากมายแบบ XML Web Service
- ◉ RESTful ออกแบบโดยมองว่า Server ให้บริการ Resource ในขณะที่ Web Services ให้บริการ Function โดยผู้ออกแบบจะออกแบบ URL เป็นช่องทางสำหรับเข้าถึง resource เช่น
 - ◉ `http://server/items/` หมายถึง item ทั้งหมดที่ server เปิดให้ใช้ได้
 - ◉ `http://server/items/1/` หมายถึง item ที่มี id เป็น 1 เท่านั้น

RESTful

- ◎ เมื่อใช้ URL เป็น resource การจัดการกับ resource ก็จะใช้ HTTP Method ที่เป็นมาตรฐานของ HTTP อยู่แล้ว
- ◎ Method ที่ใช้กันบ่อยๆ จะมี 4 methods คือ GET, PUT, POST, DELETE
- ◎ RESTful ก็จะนำ method เหล่านี้มา implement ว่า ถ้ามีการ request จาก client ว่าเป็น GET ก็จะอ่าน resource ที่ถูกระบุใน URL ส่งกลับไป แต่ถ้าเป็น POST หมายถึงการขอสร้าง object ใหม่ โดยดู detail ของ object จาก body เป็นต้น
- ◎ รูปแบบของข้อมูลมักจะใช้ JSON แต่จะใช้ XML ก็ได้
- ◎ ตัวอย่าง
 - ◎ URL http://server/items/1
Method GET
Meaning Read item id = 1
 - ◎ URL http://server/items/1
Method POST
Meaning Insert item id = 1
body {"item": { "name": "iPhone5s", "type": "Smart Phone" }}

JSON

- ◎ JSON ย่อมาจาก Java Script Object Notation เป็นมาตรฐานแบบเปิด
- ◎ ใช้เพื่ออธิบาย โครงสร้างของข้อมูลแบบเดียวกับ XML แต่ลดรูปให้สั้นกว่า ลดการใช้สัญลักษณ์ที่ไม่จำเป็น
- ◎ คิดโดย Douglas Crockford (<http://goo.gl/5gR4W>)
- ◎ ตัวอย่าง XML เปรียบเทียบกับ JSON

```
<person>
  <firstname>Barack</firstname>
  <lastname>Obama</lastname>
  <born>1961</born>
</person>
```

```
{
  "firstname" : "Barack",
  "lastname" : "Obama",
  "born" : 1961
}
```

JSON Example

➤ Name & Value

```
{ "name" : "value" }
```

➤ Array

```
{ "Regions" : ["North", "South", "Middle", "Northeastern"] }
```

➤ Anonymous Object

```
{ "FirstName" : "Steve", "LastName" : "Jobs" }
```

➤ Object

```
{"Company" : {"Name" : "G-ABLE", "Email" : "mail@g-able.com"}}
```

➤ Array of Objects

```
{ "Contacts" : [
    {"name" : "Steve Jobs", "Email" : "stevejobs@apple.com"},
    {"name" : "Mark Zuckerbergs", "Email" : "zuckerbergs@facebook.com"} ]
}
```

JSON vs. Class

```
{ "Customer" :  
  { "FirstName" : "Steve",  
    "YearOfBirth": 1950  
  }  
}
```

```
{ "Regions" : [  
  { "Name" : "Northern" },  
  { "Name" : "Southern" } ]  
}
```

```
{ "Customers" :  
  { "Contact" :  
    { "Name" : "Steve Jobs",  
      "Company" : "Apple" }  
  },  
  "Address" :  
  { "AddressDetail" : "",  
    "ZipCode" : "12345" }  
}
```

```
class Customer {  
  string FirstName;  
  int YearOfBirth;  
}
```

```
class Region {  
  string [] Name;  
}
```

```
class Customers {  
  Contact contact;  
  Address address;  
}
```

```
class Contact {  
  string Name;  
  string Company;  
}
```

```
class Address {  
  string AddressDetail;  
  string ZipCode;  
}
```

NSJSONSerialization

- เป็น class ที่ถูกเพิ่มเข้ามาใน iOS version 5 ใช้เพื่อแปลง JSON document ไปเป็น NSDictionary หรือ NSArray และแปลง Object ไปเป็น JSON Document
- Method ที่ใช้เพื่อแปลงจาก JSON คือ JSONObjectWithData:options:error: ซึ่งจะ return id อกกมา ซึ่งเราต้องรู้เองว่า id นั้นเป็น dictionary หรือ array (JSON เป็น type NSData)

```
[NSJSONSerialization JSONObjectWithData:NSData  
options:NSJSONReadingMutableContainers  
error:&error];
```

- Method ที่ใช้แปลงจาก Object เป็น JSON คือ dataWithJSONObject:option:error:

```
[NSJSONSerialization dataWithJSONObject:id  
options:NSJSONWritingPrettyPrinted  
error:&error];
```

NSJSONSerialization

⊗ Dictionary

```
{ "Name" : "Jobs" } > (key = "Name", value = "Jobs")
```

⊗ Dictionary ជំនួយ Dictionary

```
{ "Customer" : { "Name" : "Steve", "Company" : "Apple" } }
```

```
> (key = "Customer", value = Dictionary)
  >> (key = "Name", value = "Steve")
  >> (key = "Company", value = "Apple")
```

⊗ Dictionary > Array > Dictionary

```
{ "Customer" : [ { "Name" : "Steve", "Company" : "Apple" }, { "Name" : "Bill", "Company" : "Microsoft" } ] }
```

```
> (key = "Customer", value = Array)
  >> Array #1 of Dictionary
    >>> (key = "Name", value = "Steve")
    >>> (key = "Company", value = "Apple")
  >> Array #2 of Dictionary
    >>> (key = "Name", value = "Bill")
    >>> (key = "Company", value = "Microsoft")
```

NSURLConnection & NSURLSession

- ◎ การติดต่อกับ REST Server นั้น ใช้เทคโนโลยี HTTP ปกติไม่มีอะไรพิเศษ เพราะฉะนั้น ใน iOS จึงใช้ class พื้นฐานของ HTTP Client ก็สามารถทำงานกับ application ฝั่ง server ได้แล้ว
- ◎ เมื่อเวลาผ่านไป การติดต่อกับ server ด้วย HTTP Protocol ได้รับความนิยมมากขึ้น API ใน NSURLConnection เพิ่มไม่เพียงพอต่อการใช้งาน ใน iOS 7 จะมี class ใหม่เพิ่มขึ้นมา คือ NSURLSession (ไม่มี lab ใน class นี้)

NSURLConnection Class

- ⦿ ใน iOS จะมี http client เตรียมไว้ให้แล้ว คือ NSURLConnection
- ⦿ Class NSURLConnection ใช้เพื่อจัดการกับ Connection ระหว่าง client กับ server แต่ต้องอาศัย class อีก 2 ตัว คือ NSURL และ NSURLRequest
- ⦿ หลักการในการ connect คือ
 - ⦿ กำหนด URL ของ server ใน object ของ class NSURL
 - ⦿ สร้าง object NSURLRequest เพื่อกำหนด header / body ของ HTML Document
 - ⦿ สร้าง object NSURLConnection เพื่อจัดการกับการคุยกับ server ตาม configuration ที่กำหนดไว้ใน URL Request
- ⦿ การ connect ไปยัง server ทำได้ 2 แบบ คือ ใช้ method แบบ asynchronous และ implement Block กับใช้วิธี implement delegate ของ NSURLConnection

```
[NSURLConnection sendAsynchronousRequest:NSURLRequest
                                      queue: [NSOperationQueue mainQueue]
                                completionHandler:^(<params> ) { <code> } ];
```

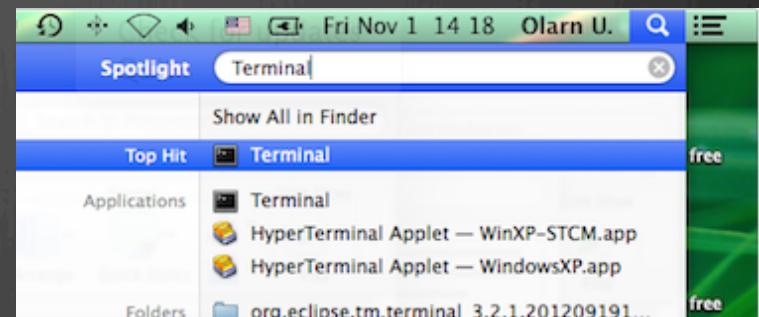
```
[NSURLConnection connectionWithRequest:request delegate:self];
```

NSURLConnection Delegate

- ⦿ Class NSURLConnection จะมี delegate method เพื่อให้เราเขียน code เมื่อเกิด event ต่างๆ method ที่ถูกใช้บ่อยๆ เช่น
 - `(void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error`
ใช้เพื่อ handle เมื่อเกิด error ในระหว่าง connect ไปยัง server
 - `(void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response`
ใช้เพื่อ handle เมื่อ server respond กลับมา (ก่อน POST ไป)
 - `(void)connection:(NSURLConnection *)connection willSendRequestForAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge`
ใช้เพื่อส่ง username และ password ไปในกรณีที่ server ต้องการ Basic Authentication
 - `(void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data`
 - `(void)connectionDidFinishLoading:(NSURLConnection *)connection`
method 2 ตัวนี้ใช้คู่กัน โดยที่ method แรกจะทยอยส่ง data กลับมาจาก server ส่วน method ที่ 2 จะทำงานเมื่อได้รับ data จาก server ครบแล้ว

Lab 1-4 : Start Web Server (1/2)

1. เปิดโปรแกรม Terminal โดย click ที่ Spotlight และพิมพ์ Terminal ในช่อง search
2. จาก Terminal ให้ cd ไปที่ folder/Resources/Day4\ -\ Lab14/ RESTServer/ (หรือ drag folder Resource และ drop ลงบน Terminal)
3. Start web server ด้วยคำสั่ง ./start.sh



```
Day4 - Lab14 — Python — 75x24
=====
Simple RESTful Test Server
=====
README
=====
Info: Browse all server files with http://localhost:8000
Info: To prepare JSON GET/POST testing...
      Just create text files with ASCII code content
      and put it into the same folder
      and call... http://localhost:8000/yourfile.json
      The server will echo-back with yourfile.json contents
Note: The example JSON is name customer.json so you can try ...
      http://localhost:8000/customer.json
Note: The content-type should be - application/json -
Press ^C to terminate or just close this terminal.
=====
@rochacbruno Python http server version 0.1 (for testing purposes only)
Serving at: http://localhost:8000
```

Lab 1-4 : Start Web Server (2/2)

4. ทดสอบว่า web server ทำงานหรือไม่ โดย
 - เปิด Safari บน Mac และ browse ไปที่ `http://localhost:8000`
 - ทดสอบ JSON โดย browse ไปที่ `http://localhost:8000/customers.json`
 - ทดสอบเปิดรูปบน web server โดย browse ไปที่
`http://localhost:8000/jobs.png`
5. เปิด iOS Emulator โดยเปิด Xcode และไปที่เมนู Xcode > Open Developer Tool > iOS Simulator
 - เปิด Safari บน iOS และ browse ไปที่ `http://localhost:8000`
 - ทดสอบ JSON โดย browse ไปที่ `http://localhost:8000/customers.json`
 - ทดสอบเปิดรูปบน web server โดย browse ไปที่
`http://localhost:8000/jobs.png`
6. ปิด web server โดยกด control + c บน keyboard

Note: ถ้า process ค้าง ให้เปิด Activity Monitor และสั่ง quit process ชื่อ “python”

Lab 2-4 : Prepare Client (1 / 13)

◎ วัตถุประสงค์

- ◎ เขียน iOS App RESTful Client เพื่อใช้แสดงข้อมูลที่ download มาจาก server

◎ ขั้นตอน

- ◎ สร้าง project
- ◎ เพิ่ม Table View และ View บน Storyboard
- ◎ วาง control ต่างๆ สำหรับแสดงข้อมูล customer
- ◎ ผูก delegate และ action
- ◎ เขียน loop เพื่อจำลอง process ที่ใช้เวลานานๆ

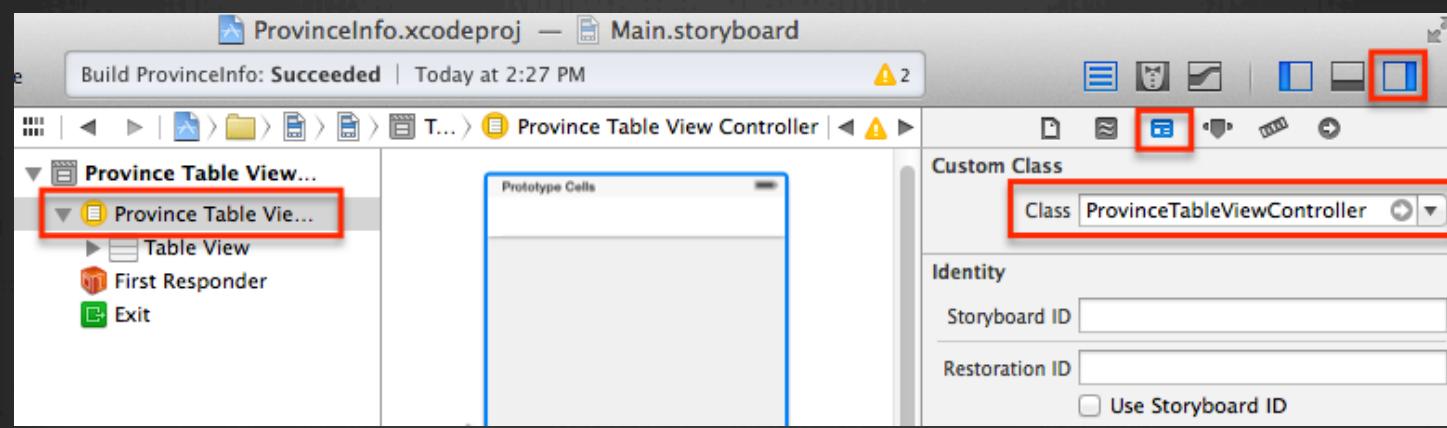
Note: ข้าม Lab 2-4 ไป โดย ใช้ code ที่เตรียมไว้ให้ใน path .. /Resources/Day4 - Lab14 /Project/

Task : Create Project (2/13)

1. จาก Xcode สร้าง project ใหม่โดยเลือก iOS > Application > Single View Application
2. ตั้งชื่อ project ว่า “iContact” และเลือก Devices เป็น iPhone
3. Click “Next” เลือก folder ที่จะ save project แล้ว click ปุ่ม “Create”
4. เปิด Main.storyboard แล้วลบ View Controller ออกจาก storyboard
5. ลบ ViewController.h และ ViewController.m ออกจาก project โดย click ขวา เลือก Delete และเลือก Move To Trash

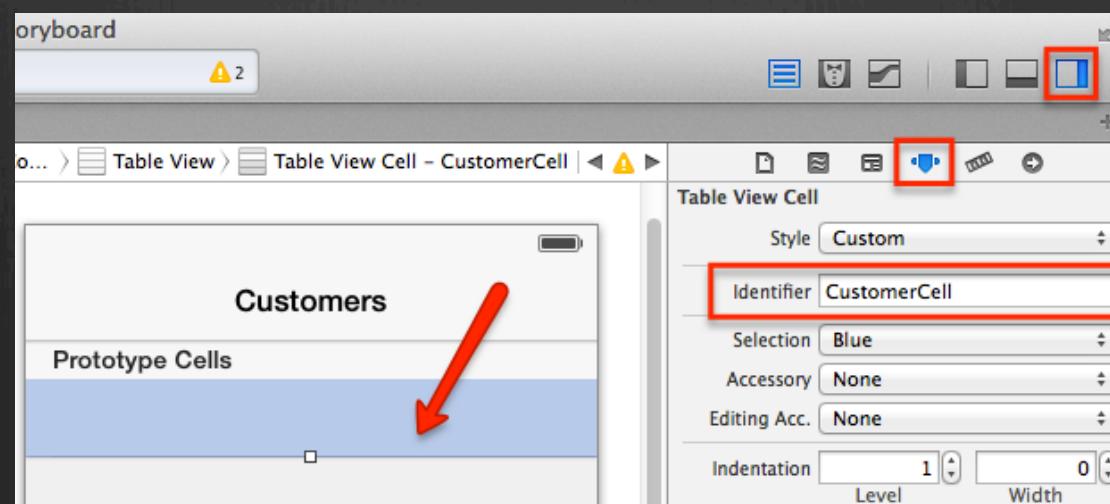
Task : UI Design – Add Table View for Customer List (3/13)

6. สร้าง class ใหม่ โดย click ขวาที่ project ใน Navigation Pane เลือก New File... > iOS > Cocoa Touch > Objective-C class และ click ปุ่ม “Next”
7. ตั้งชื่อ class ว่า “ContactTableViewController” และเลือก Subclass of เป็น “UITableViewController” (ไม่ต้องเลือก check box ทั้ง 2 ตัว) จากนั้น click ปุ่ม Next และ Create ตามลำดับ
8. เปิดไฟล์ Main.storyboard และลาก Table View Controller จาก Library Pane มาวางลงบน storyboard
9. Click ที่ table view และเปิด Identity Inspector บน Inspector Pane และเปลี่ยน class เป็น “ContactTableViewController”



Task : UI Design – Embed in Navigation (4/13)

10. เพิ่ม Navigation Controller ลงใน Table View โดย click ที่ Table View บน Storyboard และเลือกเมนู Editor > Embed In > Navigation Controller
11. Double click ที่ title ของ Navigation Pane บน Table View และตั้งชื่อ title ว่า “Customers”
12. Click ที่ Table Cell ของ Table view และเปิด Attribute Inspector ใน Inspector Pane ใส่ค่า property “Identifier” เป็น “CustomerCell”



Task : UI Design – Add View & Segue (5/13)

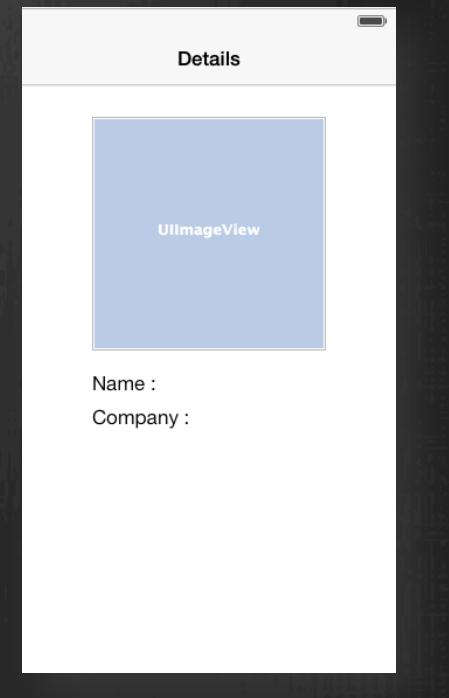
13. เพิ่ม View Controller ใหม่ลงบน storyboard
14. สร้าง Segue จาก Cell ของ Table View ไปยัง View ใหม่ โดยกด control บน keyboard ค้างไว้แล้ว drag จาก Cell ของ Table ไป drop ลงบน View และเลือก “push” ภายใต้กลุ่ม “Selection Segue”



15. Click ที่ segue แล้วกำหนด property “Identifier” ใน Attributes Inspector เป็น “segueCustomerDetail”

Task : UI Design – Decorate View for Customer Data (6/13)

16. สร้าง class ใหม่สำหรับใช้เป็น View Controller class เพื่อแสดงรายละเอียดของ Customer โดย click ขวาที่ project และเลือกเมนู New File ... > iOS > Cocoa Touch > Objective-C class > เลือก Subclass of เป็น “UIViewController” และตั้งชื่อ class ว่า “CustomerDetailViewController”
17. เปิด Main.storyboard และ click ที่ View ที่ 2 จากนั้นเปิด Identifier Inspector ใน Inspector Pane และเลือก Class เป็น “CustomerDetailViewController”
18. เพิ่ม control ลงบน View ดังนี้
 - Image View
 - Label :- Text = “Name :”
 - Label :- Text = “Company :”



Task : UI Design – Binding Outlet for Customer Detail (7/13)

19. Click เลือก View Controller และเปลี่ยน editor mode เป็น Assistant editor (ไฟล์ด้านขวาจะต้องเป็น CustomerDetailViewController.h)
20. ผูก Outlet ให้กับ Image View และ Label ทั้ง 2 ตัว โดยที่
 - Image View :-
Type = “Outlet”
Property Name = “imgCustomer”
 - Label :-
Type = “Outlet”
Property Name = “txtName”
 - Label :-
Type = “Outlet”
Property Name = “txtCompany”

Task : Coding – Create Class for Customer (8/13)

21. สร้าง class ใหม่สำหรับใช้เก็บข้อมูล Customer โดย click ขวาที่ project แล้วเลือกเมนู New File ... > iOS > Cocoa Touch > Objective-C class > เลือก Subclass of เป็น “NSObject” และตั้งชื่อ class ว่า “Customer”
22. เปิดไฟล์ “Customer.h” และเพิ่ม property ดังนี้

```
#import <Foundation/Foundation.h>

@interface Customer : NSObject

@property (nonatomic, strong) NSString * firstName;
@property (nonatomic, strong) NSString * lastName;
@property (nonatomic, strong) NSString * company;
@property (nonatomic, strong) NSString * imageName;
@property (nonatomic, strong) UIImage * image;

@end
```

Task : Coding – Implement Customer List (9/13)

23. เปิดไฟล์ “ContactTableViewController.h” เพิ่ม #import และ property ดังนี้

```
#import <UIKit/UIKit.h>
#import "CustomerDetailViewController.h"
#import "Customer.h"

@interface ContactTableViewController : UITableViewController

@property (nonatomic, strong) NSMutableArray * customers;

@end
```

24. เปิดไฟล์ “ContactTableViewController.m” แก้ code ใน method “viewDidLoad:” ดังนี้

```
- (void)viewDidLoad
{
    [super viewDidLoad];
    self.customers = [[NSMutableArray alloc] init];
}
```

Task : Coding – Implement Table View Data Source (10/13)

24. แก้ code ใน method “numberOfSectionsInTableView:” และ
“tableView:numberOfRowsInSection:” ดังนี้

```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return 1;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:
(NSInteger)section
{
    return [self.customers count];
}
```

Task : Coding – Implement Table View Data Source (11/13)

25. แก้ code ใน method “tableView:cellForRowAtIndexPath” ดังนี้

```
- (UITableViewCell *)tableView:(UITableView *)tableView
    cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *CellIdentifier = @“CustomerCell”;
    UITableViewCell *cell =
        [tableView dequeueReusableCellWithIdentifier:CellIdentifier
            forIndexPath:indexPath];

    if (!cell) {
        cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault
            reuseIdentifier:CellIdentifier];
    }

    Customer * c = [self.customers objectAtIndex:indexPath.row];
    NSString * customerName =
        [NSString stringWithFormat:@"%@ %@", c.firstName, c.lastName];
    cell.textLabel.text = customerName;

    return cell;
}
```

Task : Coding – Implement Customer View (12/13)

26. เปิดไฟล์ “CustomerDetailViewController.h” และเพิ่ม code ดังนี้

```
#import <UIKit/UIKit.h>
#import "Customer.h"

@interface CustomerDetailViewController : UIViewController

@property (weak, nonatomic) Customer * customer;

@property (weak, nonatomic) IBOutlet UIImageView *imgCustomer;
@property (weak, nonatomic) IBOutlet UILabel *txtName;
@property (weak, nonatomic) IBOutlet UILabel *txtCompany;

@end
```

27. เปิดไฟล์ “CustomerDetailViewController.h” และเพิ่ม method “ViewWillAppear:” ดังนี้

```
- (void)viewWillAppear:(BOOL)animated
{
    if (self.customer) {
        self.txtName.text =
            [NSString stringWithFormat:@"Name : %@ %@", self.customer.firstName,
                                   self.customer.lastName];
        self.txtCompany.text =
            [NSString stringWithFormat:@"Company : %@", self.customer.company];
        if (self.customer.image)
            self.imgCustomer.image = self.customer.image;
    }
}
```

Task : Coding – Implement Customer View (13/13)

28. เปิดไฟล์ “ContactTableViewController.m” ลงมาด้านล่างสุดแล้วเอา comment ของ method “prepareForSegue:sender:” ออก จากนั้นเพิ่ม code ดังนี้

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
{
    if ([segue.identifier isEqualToString:@"segueCustomerDetail"]) {
        CustomerDetailViewController *
            detailView = (CustomerDetailViewController *)
                [segue destinationViewController];

        NSIndexPath * indexPath = self.tableViewIndexPathForSelectedRow;
        detailView.customer = [self.customers objectAtIndex:indexPath.row];
    }
}
```

Lab 3-4 : Implement RESTful Client (1 / 7)

◎ วัตถุประสงค์

- ◎ เพื่อให้เข้าใจการเรียกข้อมูล JSON จาก RESTful web service ด้วย class NSURLConnection

◎ ขั้นตอน

- ◎ เพิ่ม code “Pull-down to Refresh” บน Table เพื่อ download ข้อมูล
- ◎ เพิ่ม code การ download ข้อมูล JSON และแปลงเป็น object
- ◎ เพิ่ม code การ download รูปเมื่อแสดงข้อมูลในหน้า detail

Task : Coding – Pull Down to Refresh (2/7)

1. เปิดไฟล์ “ContactTableViewController.m” เพิ่ม method ใหม่เพื่อ download ข้อมูลจาก server และหยุด animation การ refresh ข้อมูลของ Table View

```
- (void)refreshCustomers
{
    [self.refreshControl endRefreshing];
}
```

2. เพิ่ม code ใน method “viewDidLoad:” เพื่อเปิด feature “Pull down to Refresh” ให้กับ Table View

```
- (void)viewDidLoad
{
    [super viewDidLoad];
    self.customers = [[NSMutableArray alloc] init];

    UIRefreshControl * refreshControl = [[UIRefreshControl alloc] init];
    [refreshControl addTarget:self
                      action:@selector(refreshCustomers)
            forControlEvents:UIControlEventValueChanged];
    self.refreshControl = refreshControl;
}
```

3. ทดลอง run โปรแกรมดู จะเห็นว่าสามารถดึง Table View ลงมาเพื่อ refresh ข้อมูลได้แล้ว

Task : Coding – Download JSON (3/7)

4. จาก method “refreshCustomers” ลบ code เดิมออก (ย้ายไปอยู่ใน code ของข้อ 5.)
แล้วเพิ่ม code เพื่อสร้าง NSURLConnection และ download ข้อมูล JSON

```
- (void)refreshCustomers
{
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];

    NSURL * url = [NSURL URLWithString:@"http://localhost:8000/customers.json"];
    NSURLRequest * request = [NSURLRequest requestWithURL:url];

    [NSURLConnection sendAsynchronousRequest:request
                                           queue:[NSOperationQueue mainQueue]
                                         completionHandler:^(NSURLResponse *response,
                                                             NSData *data,
                                                             NSError *connectionError) {
        // ...
        // เพิ่ม code ในข้อ 5.
        // ...
    }];
}
```

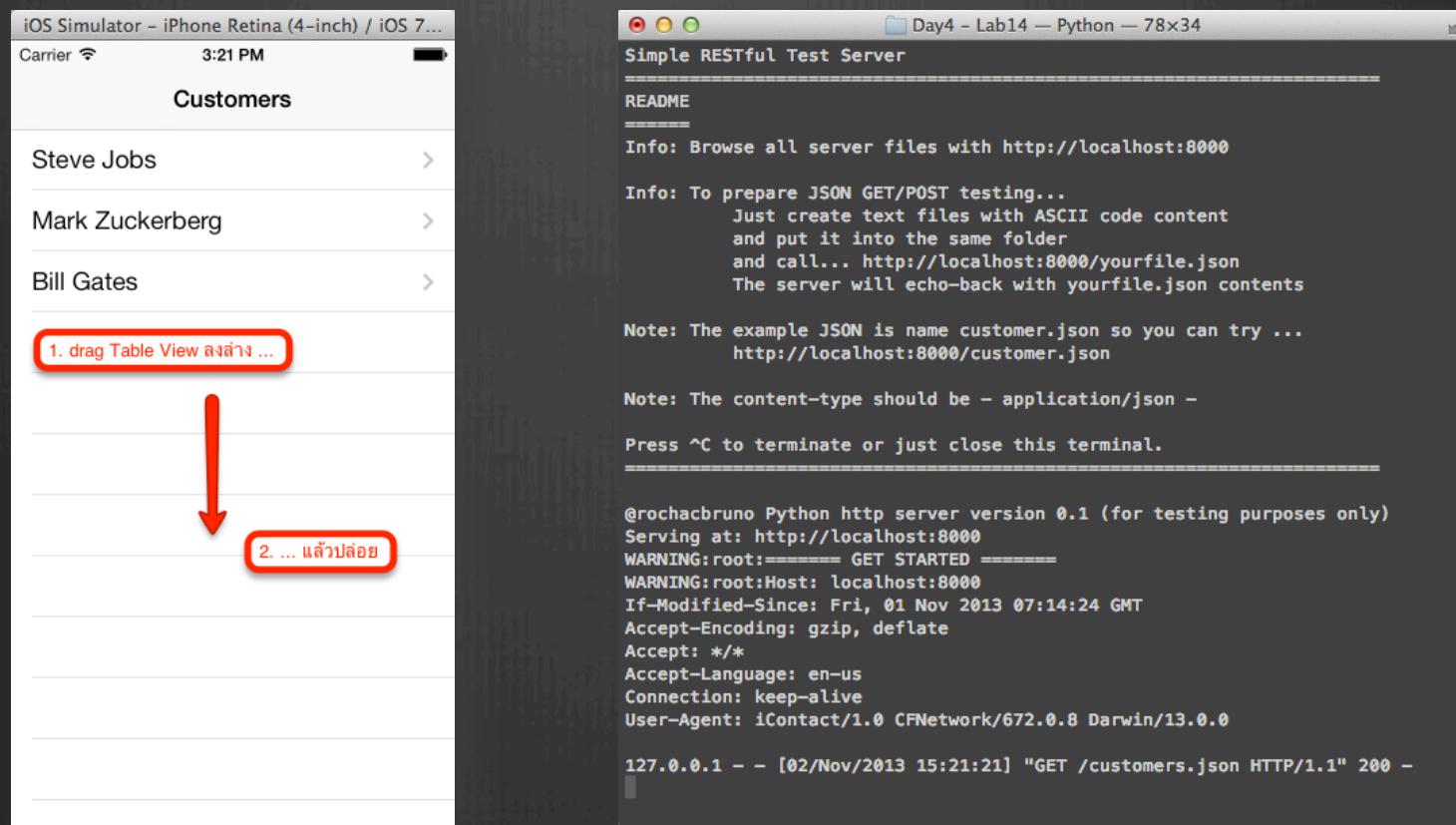
Task : Coding – Convert JSON to Objects (4/7)

5. เพิ่ม code ภายใต้ Block code ที่เว้นไว้ในข้อ 4. เพื่อแปลง JSON Document ไปเป็น NSDictionary และเอาค่าจาก dictionary ไปสร้าง object Customer

```
// ...  
  
[[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];  
[self.refreshControl endRefreshing];  
  
NSError * error;  
NSDictionary *jsonDoc = [NSJSONSerialization JSONObjectWithData:data  
options:NSJSONReadingMutableContainers  
error:&error];  
  
if (!error) {  
    [self.customers removeAllObjects];  
    NSArray * customerArray = [jsonDoc objectForKey:@"customers"];  
  
    for (NSDictionary * eachProperty in customerArray) {  
        Customer * c = [[Customer alloc] init];  
        c.firstName = [eachProperty objectForKey:@"firstname"];  
        c.lastName = [eachProperty objectForKey:@"lastname"];  
        c.company = [eachProperty objectForKey:@"company"];  
        c.imageName = [eachProperty objectForKey:@"image"];  
        [self.customers addObject:c];  
    }  
    [self.tableView reloadData];  
}  
  
// ...
```

Task : Run & Test (5/7)

6. Run โปรแกรมเพื่อทดสอบผลลัพธ์



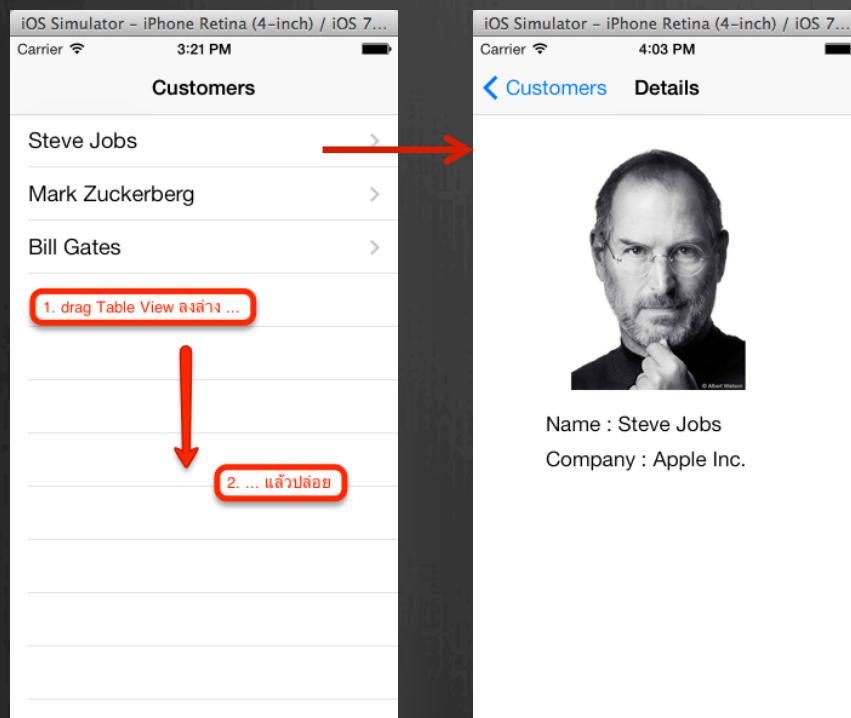
Task : Coding – Download Image (6/7)

7. เปิดไฟล์ “CustomerDetailViewController.m” และ code ใน method “viewWillAppear:” เพื่อ download รูปและแสดงผล

```
- (void)viewWillAppear:(BOOL)animated
{
    if (self.customer) {
        self.txtName.text = [NSString stringWithFormat:@"Name : %@ %@", self.customer.firstName,
                             self.customer.lastName];
        self.txtCompany.text = [NSString stringWithFormat:@"Company : %@", self.customer.company];
        if (self.customer.image)
            self.imgCustomer.image = self.customer.image;
    } else {
        [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];
        NSString * urlStr = [NSString stringWithFormat:@"http://localhost:8000/%@", self.customer.imageName];
        NSURL * url = [NSURL URLWithString:urlStr];
        NSURLRequest * request = [NSURLRequest requestWithURL:url];
        [NSURLConnection sendAsynchronousRequest:request
                                              queue:[NSOperationQueue mainQueue]
                                         completionHandler:^(NSURLResponse *response, NSData *data, NSError *connectionError) {
            [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];
            self.customer.image = [UIImage imageWithData:data];
            self.imgCustomer.image = self.customer.image;
        }];
    }
}
```

Task : Run & Test (7/7)

6. Run โปรแกรมเพื่อทดสอบผลลัพธ์



```

Day4 - Lab14 — Python — 75x34
=====
Simple RESTful Test Server
=====
README
=====
Info: Browse all server files with http://localhost:8000

Info: To prepare JSON GET/POST testing...
      Just create text files with ASCII code content
      and put it into the same folder
      and call... http://localhost:8000/yourfile.json
      The server will echo-back with yourfile.json contents

Note: The example JSON is name customer.json so you can try ...
      http://localhost:8000/customer.json

Note: The content-type should be - application/json -
Press ^C to terminate or just close this terminal.

=====
@rochacbruno Python http server version 0.1 (for testing purposes only)
Serving at: http://localhost:8000
WARNING:root:===== GET STARTED =====
WARNING:root:Host: localhost:8000
If-Modified-Since: Fri, 01 Nov 2013 07:11:05 GMT
Accept-Encoding: gzip, deflate
Accept: /*
Accept-Language: en-us
Connection: keep-alive
User-Agent: iContact/1.0 CFNetwork/672.0.8 Darwin/13.0.0

127.0.0.1 -- [04/Nov/2013 11:13:01] "GET /zuck.png HTTP/1.1" 200 -

```

Lab 4-4 : POST (1/7)

◎ วัตถุประสงค์

- ◎ เพื่อให้เข้าใจการใช้ HTTP POST เพื่อ update ข้อมูลกลับไปยัง server

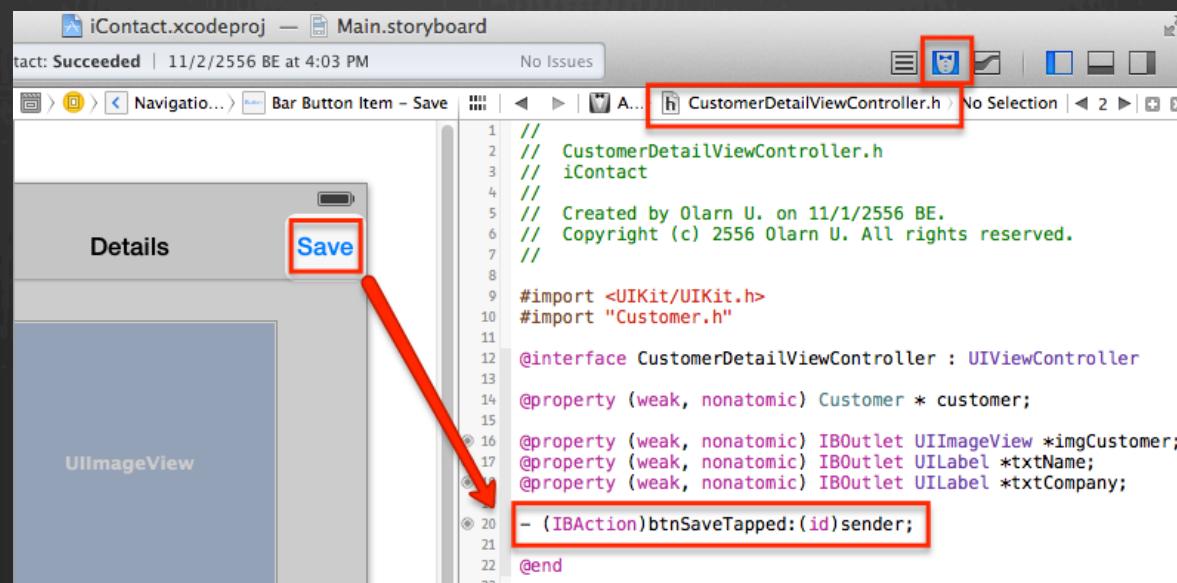
◎ ขั้นตอน

- ◎ เพิ่มปุ่ม Save ในหน้า Detail
- ◎ เขียน code เพื่อกำหนดค่าของ HTTP Header
- ◎ แปลง Object เป็น JSON Data
- ◎ Upload กลับไปยัง server

Note: web server ที่ใช้ใน lab เป็นแค่ server จำลอง สามารถรับ POST message ได้แต่ไม่มีอะไร update ที่ server จริง

Task : UI Design & Binding Action (2/7)

1. เพิ่ม control “Bar Button Item” ลงบน Navigation Bar ของ view Detail แล้วเปลี่ยน property “Identifier” (ใน Attribute inspector) เป็น “Save”
2. เปลี่ยน editor mode เป็น Assistant Editor แล้วผูก Action กับไฟล์ “CustomerDetailViewController.h” ตั้งชื่อว่า “btnSaveTapped”



Task : Coding – Delegate Handler Declaration (3/7)

3. เพิ่ม delegate declaration ใน @interface ของไฟล์ “CustomerDetailViewController.h” เพื่อ handle delegate ของ NSURLConnection และเพิ่ม property “returnedData” เพื่อใช้รับ data จาก server

```
#import <UIKit/UIKit.h>
#import "Customer.h"

@interface CustomerDetailViewController : UIViewController
    <NSURLConnectionDelegate, NSURLConnectionDataDelegate>

@property (weak, nonatomic) Customer *customer;

@property (nonatomic, strong) NSMutableData * returnedData;

@property (weak, nonatomic) IBOutlet UIImageView *imgCustomer;
@property (weak, nonatomic) IBOutlet UILabel *txtName;
@property (weak, nonatomic) IBOutlet UILabel *txtCompany;

- (IBAction)btnSaveTapped:(id)sender;

@end
```

Note: ใน iOS 6 ลงไปเรา implement delegate แต่ class เดียว คือ NSURLConnectionDelegate แต่ใน iOS 7 method ส่วนหนึ่งถูกย้ายไปอยู่ใน NSURLConnectionDataDelegate จึงต้องเพิ่ม delegate handler เป็น 2 ตัว

Task : Coding – Implement Delegate Handler (4/7)

4. เพิ่ม method “connection:didFailWithError:”, “connection:didReceiveResponse:” และ “connection:willSendRequestForAuthenticationChallenge:” เพื่อ handle delegate ของ NSURLConnection

```
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
{
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];
    NSLog(@"Connection Error");
}

- (void)connection:(NSURLConnection *)connection
didReceiveResponse:(NSURLResponse *)response
{
    NSLog(@"Respond from server");
}

- (void)connection:(NSURLConnection *)connection
willSendRequestForAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge
{
    // Replace username and password with basic user authentication if requires.
    NSURLCredential *credential = [NSURLCredential credentialWithUser:@"user"
                                                               password:@"password"
                                                               persistence:NSURLCredentialPersistenceForSession];
    [[challenge sender] useCredential:credential forAuthenticationChallenge:challenge];
}
```

Task : Coding – Implement Delegate Handler (5/7)

5. เพิ่ม method “connection:didReceiveData:” เพื่อรับข้อมูลจาก server

```
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
{
    if (!self.returnedData)
        self.returnedData = [[NSMutableData alloc] init];

    [self.returnedData appendData:data];
}
```

6. เพิ่ม method “connectionDidFinishLoading:” เพื่อเพื่อแสดงข้อมูลที่ได้จาก server

```
- (void)connectionDidFinishLoading:(NSURLConnection *)connection
{
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];

    NSError *error = nil;
    if (self.returnedData)
    {
        NSDictionary * result = [NSJSONSerialization JSONObjectWithData:self.returnedData
                                                               options:NSJSONReadingMutableLeaves
                                                               error:&error];
        [[[UIAlertView alloc] initWithTitle:@"Success"
                                     message:[result objectForKey:@"result"]
                                     delegate:nil
                                     cancelButtonTitle:@"OK"
                                     otherButtonTitles:nil, nil] show];
    }
}
```

Task : Coding – POST to Server (5/6)

7. เพิ่ม code ใน method “btnSaveTapped:” เพื่อ POST ข้อมูลกลับ server

```
- (void)btnSaveTapped:(id)sender
{
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];

    NSDictionary * customerDictionary = [NSDictionary dictionaryWithObjectsAndKeys:
                                         self.customer.firstName, @"firstName",
                                         self.customer.lastName, @"lastName",
                                         self.customer.company, @"company", nil];
    NSError * error;
    NSData * customerData = [NSJSONSerialization dataWithJSONObject:customerDictionary
                                                               options:NSJSONWritingPrettyPrinted
                                                               error:&error];

    NSURL *url = [NSURL URLWithString:@"http://localhost:8000/result.json"];
    NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
                                                               cachePolicy:NSURLRequestUseProtocolCachePolicy
                                                               timeoutInterval:30];

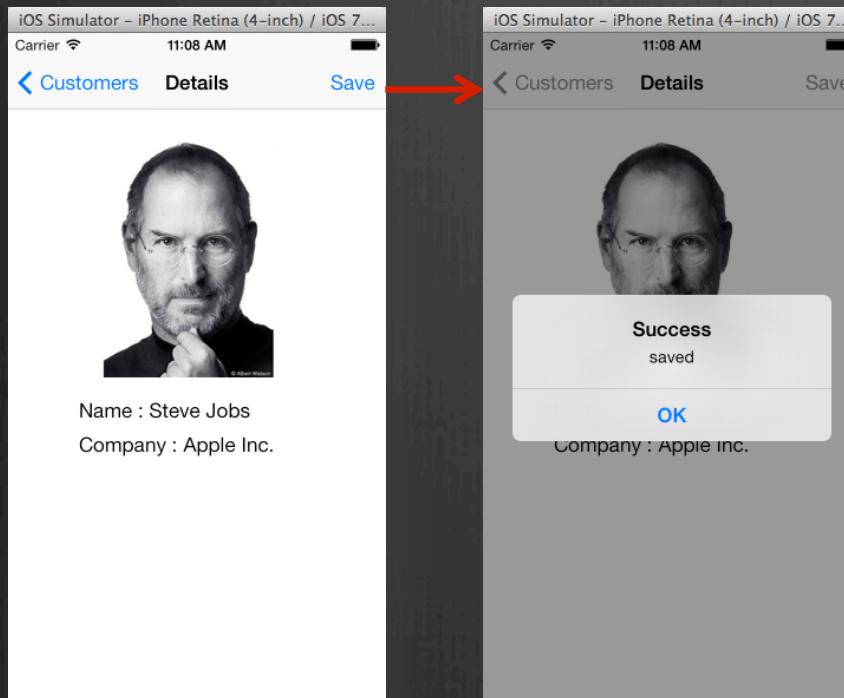
    [request setValue:@"application/json; charset=utf-8" forHTTPHeaderField:@"Content-Type"];
    [request setValue:@"application/json" forHTTPHeaderField:@"Accept"];
    [request setHTTPMethod:@"POST"];
    [request setHTTPBody:customerData];

    self.returnedData = nil;

    [NSURLConnection connectionWithRequest:request delegate:self];
}
```

Task : Run & Test (7/7)

8. Run โปรแกรมเพื่อดูผลลัพธ์



```
Day4 - Lab14 — Python — 75x36
=====
Simple RESTful Test Server
=====
README
=====
Info: Browse all server files with http://localhost:8000
Info: To prepare JSON GET/POST testing...
      Just create text files with ASCII code content
      and put it into the same folder
      and call... http://localhost:8000/yourfile.json
      The server will echo-back with yourfile.json contents
Note: The example JSON is name customer.json so you can try ...
      http://localhost:8000/customer.json
Note: The content-type should be - application/json -
Press ^C to terminate or just close this terminal.
=====

@rochacbruno Python http server version 0.1 (for testing purposes only)
Serving at: http://localhost:8000
WARNING:root:===== POST STARTED =====
WARNING:root:Host: localhost:8000
Accept-Encoding: gzip, deflate
Content-Type: application/json; charset=utf-8
Content-Length: 78
Accept-Language: en-us
Accept: application/json
Connection: keep-alive
User-Agent: iContact/1.0 CFNetwork/672.0.8 Darwin/13.0.0
WARNING:root:===== POST VALUES =====
127.0.0.1 - - [04/Nov/2013 11:09:58] "POST /result.json HTTP/1.1" 200 -
```