iPhone与iPad应用开发课程
精通iOS开发

第十讲 云端应用

主讲人：关东升

eorient@sina.com

主要知识点

- GET请求
- XML解析
- JSON解析
- POST请求

GET请求

- 通过一个第三方提供的云服务，查询IP归属地：
- http://www.youdao.com/smartresult-xml/search.s?type=ip&q=218.241.121.186
- 它的返回格式是xml

```
<?xml version="1.0" encoding="gob"?>
<smartresult>
<product type="ip"><ip>218.241.121.186</ip>
<location>北京市 电信通</location></product>
</smartresult>
```

CSSimpleXML

点击按钮事件

```
-(IBAction)onClickButton:(id)sender {

        //NSString *str = [[NSString alloc] initWithFormat:@"Hello. %@",
txtField.text];

        NSString *strURL = [NSString stringWithFormat:@"http://
www.youdao.com/smartresult-xml/search.s?type=ip&q=%@", txtField.text];
        NSURL *url = [NSURL URLWithString:strURL];

        NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];

        NSURLConnection *connection = [[NSURLConnection alloc]
           initWithRequest:request
                  delegate:self];

        [connection release];
        [request release];

        [activityIndicator startAnimating];

}
```

解释

- 定义NSURLConnection的委托：
- NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];
- NSURLConnection *connection = [[NSURLConnection alloc] initWithRequest:request delegate:self];
- 委托（delegate）是一种事件处理机制，当满足条件时候触发。delegate:self说明是委托当前对象处理事件，我们需要实现它们回调方法。

NSURLConnection 回调方法

- -(void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data

请求成功，并且接收数据。

- -(void) connection:(NSURLConnection *)connection didFailWithError: (NSError *)error

请求成功，但是加载数据出现异常。

- - (void) connectionDidFinishLoading: (NSURLConnection*) connection

加载数据成功，在connection:didReceiveData方法之后执行。

接收数据处理

```
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data {

        //upateData = [[NSData alloc] initWithData:data];
        //默认对于中文的支持不好
        NSStringEncoding enc = CFStringConvertEncodingToNSStringEncoding
(kCFStringEncodingGB_18030_2000);
        NSString *gbkNSString = [[NSString alloc] initWithData:data encoding: enc];
        //如果是非UTF－8  NSXMLParser会报错。
        xmlString = [[NSString alloc] initWithString:[gbkNSString
 stringByReplacingOccurrencesOfString:@"<?xml version=\"1.0\" encoding=\"gbk\"?>"

withString:@"<?xml version=\"1.0\" encoding=\"utf-8\"?>"]];

        NSLog(@"%@", xmlString);
        //NSLog(@"%@", utf8NSString);
        [gbkNSString release];

}
```

解释

- iPhone SDK提供的XML解析类只能解析utf-8编码，如果从服务器返回的xml编码是gbk等，要转换成utf-8再开始解析。

XML解析

- 关于iPhone发送GET请求，就是通过NSURLRequest和NSURLConnection两个类实现的。
- 在众多的回调方法。解析XML是在connectionDidFinishLoading:方法开始的。
- 解析XML文件也是要通过XML回调方式实现解析处理的。
- NSXMLParser，是iPhone解析XML SDK工具类。NSXMLParser采用SAX方式而不是DOM方式解析，SAX是基于事件触发的解析方式，解析器从上到下遍历xml文档，遇到开始标签、结束标签、文档开始、文档结束和字符串都会触发事件。

解析开始处理

```
- (void) connectionDidFinishLoading: (NSURLConnection*) connection {
        [activityIndicator stopAnimating];

        NSLog(@"%@", xmlString);

        //开始解析XML
        NSXMLParser *ipParser = [[NSXMLParser alloc] initWithData:[xmlString
dataUsingEncoding:NSUTF8StringEncoding]];
        ipParser.delegate = self;
        [ipParser parse];
        [ipParser release];
}
```

NSXMLParser回调方法

- - (void)parserDidStartDocument:(NSXMLParser *)parser
- 文档开始的时候触发
- - (void)parser:(NSXMLParser *)parser parseErrorOccurred:(NSError *)parseError
- 文档出错的时候触发
- (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName namespaceURI:(NSString *)namespaceURI qualifiedName:(NSString *)qualifiedName attributes:(NSDictionary *)attributeDict
- 遇到一个开始标签时候触发。

NSXMLParser回调方法

- - (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string
- 遇到字符串时候触发
- - (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementNamenamespaceURI:(NSString *)namespaceURIqualifiedName:(NSString *)qName
- 遇到结束标签时候出发。
- - (void)parserDidEndDocument:(NSXMLParser *)parser
- 遇到文档结束时候触发。

文档开始的回调方法

- 这个方法在解析过程中只调运一次，一般在这个方法中进行有关解析的初始化处理。

```
- (void)parserDidStartDocument:(NSXMLParser *)parser {
        info = [[NSMutableDictionary alloc]
         initWithCapacity: 1];
}
```

文档出错回调方法

```
- (void)parser:(NSXMLParser *)parser parseErrorOccurred:(NSError *)parseError {
        UIAlertView *errorAlert =
        [[UIAlertView alloc]
                initWithTitle: [parseError localizedDescription]
                        message: [parseError localizedFailureReason]
                        delegate:nil
                        cancelButtonTitle:@"OK"
                        otherButtonTitles:nil];
        [errorAlert show];
        [errorAlert release];
}
```

遇到开始标签回调方法

- 参数elementName是标签的名字，attributeDict是属性列表，namespaceURI 是命名空间，如果有命名空间qualifiedName是指定的前缀名。

```
- (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName
        namespaceURI:(NSString *)namespaceURI
        qualifiedName:(NSString *)qualifiedName
        attributes:(NSDictionary *)attributeDict  {
        NSLog(@"value: %@\n", elementName);
        currentTagName = elementName;
}
```

遇到字符串回调方法

```
- (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string {
        NSLog(@"value: %@\n", string);

        string = [string stringByReplacingOccurrencesOfString:@"\n" withString:@""];

        if ([currentTagName isEqualToString:@"ip"]) {
                if (![string isEqualToString:@""]) {
                        [info setValue: string forKey: currentTagName];
                }
        } else if ([currentTagName isEqualToString:@"location"]) {
                if (![string isEqualToString:@""]) {
                        [info setValue: string forKey: currentTagName];
                }
        }
}
```

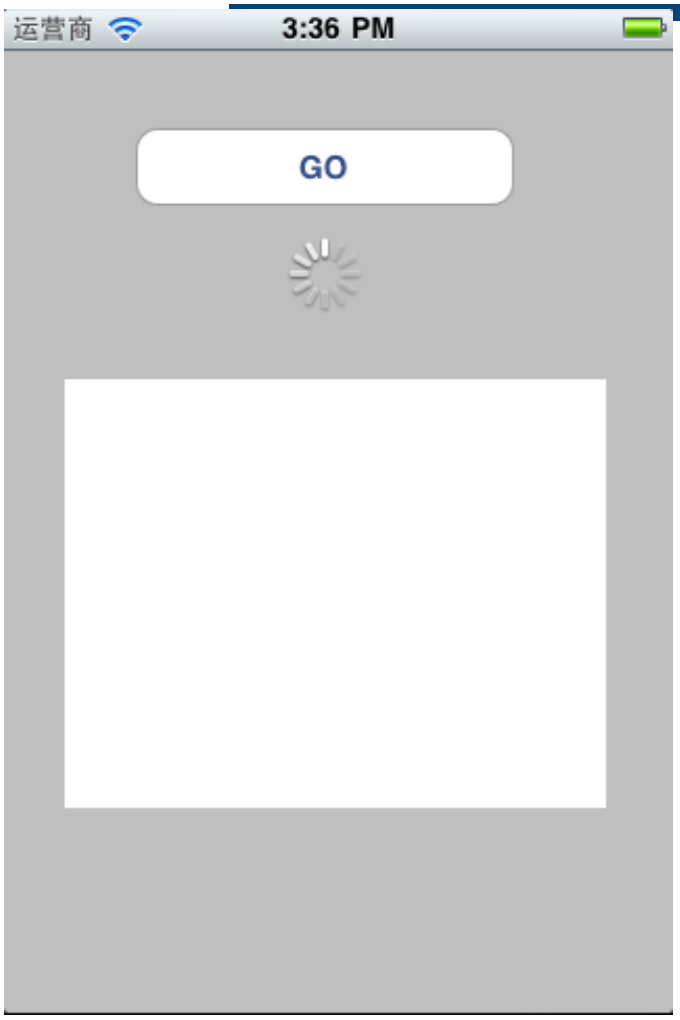
遇到结束标签回调方法

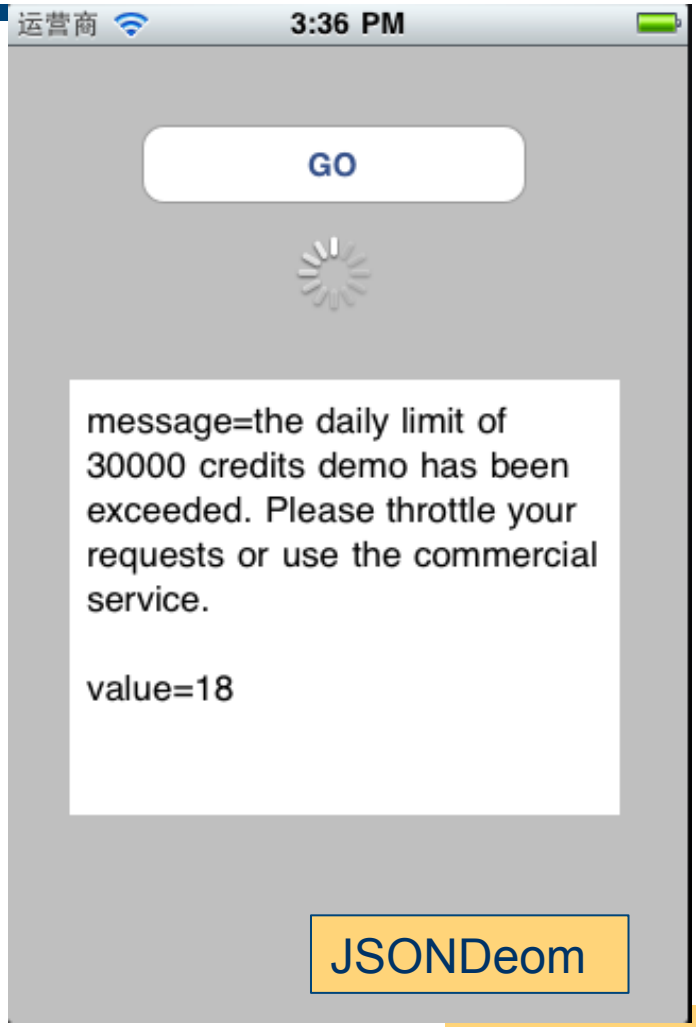
```
- (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementName
                        namespaceURI:(NSString *)namespaceURI
                        qualifiedName:(NSString *)qName {

}
```

遇到结束文档回调方法

```
- (void)parserDidEndDocument:(NSXMLParser *)parser {

        NSMutableString *outstring = [[NSMutableString alloc] initWithCapacity: 1];

        for (id key in info) {
                [outstring appendFormat: @"%@: %@\n", key, [info objectForKey:key]];
        }

        updateView.text = outstring;

        [outstring release];
        [xmlString release];
}
```

JSON解析


运营商
3:36 PM
GO


运营商
3:36 PM
GO
message=the daily limit of 30000 credits demo has been exceeded. Please throttle your requests or use the commercial service.
value=18
JSONDeom

JSON解析

- JSON格式不再介绍！
- 我们安排一个案例从第三方获得ip地址信息。
- http://www.geonames.org/export/ws-overview.html
- 获得JSON：

```
{
   status =     {
      message = "the daily limit of 30000 credits demo has been exceeded.
Please throttle your requests or use the commercial service.";
      value = 18;
   };
}
```

JSON解析API

- iPhone SDK没有提供JSON解析API，可以使用第三方的API类库json-framework，下载地址：
- https://github.com/stig/json-framework/
- 把Classes/JSON/下面的类拷贝到我们的工程的Classes目录下面，右键添加存在的类文件。

实现回调方法

- - (void)connection:(NSURLConnection *) connection didReceiveData:(NSData *)data
- - (void) connectionDidFinishLoading: (NSURLConnection*) connection
- -(void) connection:(NSURLConnection *) connection didFailWithError: (NSError *)error

connection:didReceiveData:

```
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)
data {
        outString = [[NSString alloc] initWithData:data
            encoding: NSUTF8StringEncoding];
        NSLog(@"%@", outString);
}
```

connectionDidFinishLoading:

- outString的 JSONValue消息获得NSMutableDictionary，JSON api中提供了NSString的分类（Catelog）

```
- (void) connectionDidFinishLoading: (NSURLConnection*) connection {
        [activityIndicatorView stopAnimating];

    NSMutableDictionary *jsonoObj = [outString JSONValue];
    NSLog(@"%@", jsonoObj);
    NSMutableDictionary *jsonoSubObj = [jsonoObj objectForKey:@"status"];

    NSString *text = [[NSString alloc] initWithFormat:@"message=%@\n\nvalue=%@",[jsonoSubObj
objectForKey:@"message"], [jsonoSubObj objectForKey:@"value"]];
    textView.text = text;
    [text release];
        [outString release];
}
```

文档出错回调方法

```
- (void)parser:(NSXMLParser *)parser parseErrorOccurred:(NSError *)parseError {
        UIAlertView *errorAlert =
        [[UIAlertView alloc]
                initWithTitle: [parseError localizedDescription]
                        message: [parseError localizedFailureReason]
                        delegate:nil
                        cancelButtonTitle:@"OK"
                        otherButtonTitles:nil];
        [errorAlert show];
        [errorAlert release];
}
```

点击按钮事件

```
-(IBAction)go:(id)sender
{
    NSString *strurl = @"http://api.geonames.org/findNearByWeatherJSON?
lat=43&lng=-2&username=demo";
        NSURL *url = [NSURL URLWithString:strurl];

        NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];

        NSURLConnection *connection = [[NSURLConnection alloc]

initWithRequest:request

delegate:self];

        [connection release];
        [request release];

        [activityIndicatorView startAnimating];

}
```

POST请求

- 为了学习iPhone的POST请求，安排案例如下：
- 在画面中输入用户名和密码，然后以POST方式提交数据到http://www.51work6.com/a.php。

CSSimplePOST

NSMutableURLRequest

- POST请求与GET不同，不使用的NSURLRequest，而是使用NSMutableURLRequest类，这是一个可变的NSURLRequest类。

```
-(IBAction)login:(id)sender {

        NSString *post = [NSString stringWithFormat:@"name=%@&password=%@",
                                                txtUserName.text, txtPwd.text];

        NSData *postData = [post dataUsingEncoding:NSUTF8StringEncoding];

        NSURL *webServiceURL = [NSURL URLWithString:@"http://www.51work6.com/a.php"];
        NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:webServiceURL];
        [request setHTTPMethod:@"POST"];
        [request setHTTPBody:postData];
        NSURLConnection *connection = [[NSURLConnection alloc]
         initWithRequest:request delegate:self];
        if (!connection) {
                NSLog(@"Failed to submit request");
        } else {
                NSLog(@"Request submitted");
        }
        [connection release];

        [activityIndicator startAnimating];
}
```

解释

- POST参数是以一个字符串方式传递：
  "name=tony&password=123”
- [request setHTTPMethod:@"POST"];
- 知道请求方法为POST方法，但是要注意POST必须大写。
- [request setHTTPBody:postData];该语句是将要提交的数据放到请求体中。

connection:didReceiveData:

```
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)
  data
{
        resultString = [[NSString alloc] initWithData:data
                                        encoding:NSUTF8StringEncoding];
}
```

connectionDidFinishLoading:

```
- (void)connectionDidFinishLoading:(NSURLConnection *)theConnection {
        NSString *message;
        if ([resultString isEqual:@"1"]) {
                message = [[NSString alloc] initWithFormat:@"%@", @"登录成功！ "];
        } else {
                message = [[NSString alloc] initWithFormat:@"%@", @"登录失败！ "];
        }
        UIAlertView *alert = [[UIAlertView alloc]
                                        initWithTitle: @"登录结果"
                                        message: message
                                         delegate:nil
                                          cancelButtonTitle:@"OK"
                                          otherButtonTitles:nil];
        [alert show];
        [alert release];
        [resultString release];

        [activityIndicator stopAnimating];
}
```

文档出错回调方法

```
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)
error
{
    UIAlertView *errorAlert =     [[UIAlertView alloc]
                                initWithTitle: [error localizedDescription]
                                message: [error localizedFailureReason]
                                delegate:nil
                                cancelButtonTitle:@"OK"
                                otherButtonTitles:nil];
            [errorAlert show];
            [errorAlert release];

}
```