

Ionic Study

Day 14

오늘 할 것들

- AlertController 살펴보기
- 서버 restAPI 호출해보기
 - HTTP GET, POST 호출
 - 오픈API 활용해보기

AlertController

- git clone <https://github.com/KimTibber/ionic-sample-grid>
- cd ionic-sample-grid
- npm install

AlertController

- PromptAlert
 - 간단한 입력을 받을 수 있는 알럿
 - input 속성에 배열로 값 전달
 - name: 입력값을 받을 변수
 - placeholder: 입력상자에 보여질 문구
 - type: 입력 상자의 타입
 - 입력받은 값은 handler 속성의 람다함수에서 data 매개변수로 값을 받아올 수 있음.

AlertController

- src/pages/home/home.html

```
<div>
  <p>아나타노 나마에와... {{ userName }}</p>
  <button ion-button (click)="getUserName()">What your name?</button>
</div>
```

AlertController

- src/pages/home/home.ts

- userName 멤버변수 선언

```
9  export class HomePage {  
10 |   userName;
```

AlertController

- src/pages/home/home.ts

```
70 |     getUserName() {
71 |       let prompt = this.alertCtrl.create({
72 |         title: '왓유얼네임?',
73 |         inputs: [
74 |           {
75 |             name: 'userName',
76 |             placeholder: '이름 입력'
77 |           }
78 |         ],
79 |         buttons: [
80 |           {
81 |             text: '닫기',
82 |             role: 'cancel'
83 |           },
84 |           {
85 |             text: '확인',
86 |             handler: data => {
87 |               this.userName = data.userName;
88 |             }
89 |           }
90 |         ]
91 |       });
92 |       prompt.present();
93 |     }
94 |   }
```

restful API

- 클라이언트와 서버의 통신은 어떻게?
- API를 이용하여 통신
- 서버에 요청 => 서버의 응답 처리 순으로
진행.

restful API

- GET와 POST
 - HTTP 프로토콜에서 웹 서버에 요청을 보내는 방법
 - 요청 데이터를 주소(URL)에 실어서 보내느냐? 요청 바디에 숨겨서 보내느냐의 차이.
 - restful API에서는 통상적으로 취득을 위한 요청은 GET로, 정보의 입력을 위한 요청 및 GET로 보내기 어려운 데이터(개인 정보등)은 POST로 보냄.

restful API

- ionic에서의 API 사용
 - AJAX를 이용하여 서버와 통신
 - 비동기 자바스크립트 & XML 의 줄임말
 - 페이지 갱신 없이 DOM과 상호작용이 가능
 - httpClient 모듈을 이용하여 AJAX 통신을 함.

restful API

- HttpClientModule
 - app.module.ts에 import
- 오픈 API 사용해보기
 - 오픈 API는 API 제공사에서 대가 없이 공개적으로 제공하는 API
 - ex) 네이버 검색 API, 행정안전부 새주소 API등

restful API

- 오픈 API 사용해보기
 - httpClientModule를 이용하여 네이버 검색 API 긁어와보기를 진행
 - 검색어 입력부와 검색 결과 노출부 필요.
- ionic start rest1 blank

네이버 검색 해보기

검색어 치킨샐러드

검색해보기

치킨샐러드
✓ 뷔어로 보기 이자까야 2015.01.12 10:5 >

갈릭 허브 치킨샐러드 | 파스퇴르(허브 치킨샐러드 0 담백하게 즐기 >

치킨샐러드
✓ 뷔어로 보기 일식 2012.01.20 15:49

교촌치킨
치킨 전문점, 메뉴, 영양 정보, 전국 배송 >

BBQ 치킨
치킨 프랜차이즈, 22년 역사 BBC >

사바사바치킨&비어
생맥주 전문점, 파닭치킨, 오븐구이 치킨 >

치킨샐러드
요리그룹 '치킨샐러드'(으)로 묶인 >

restful API

- 오픈 API 사용해보기
 - API 사용 가이드 참조 필요
 - <https://developers.naver.com/docs/search/web/>

restful API

- src/app/app.module.ts

```
1 | import { HttpClientModule } from '@angular/common/http';
```

```
16    imports: [
17      BrowserModule,
18      HttpClientModule,
19      IonicModule.forRoot(MyApp)
20    ],
```

restful API

- src/pages/home/home.html

```
1  <ion-header>
2    <ion-navbar>
3      <ion-title>
4        네이버 검색 해보기
5      </ion-title>
6    </ion-navbar>
7  </ion-header>
8
9  <ion-content padding>
10   <div>
11     <ion-list>
12       <ion-item>
13         <ion-label fixed>검색어</ion-label>
14         <ion-input type="text" [(ngModel)]>
15           ="searchQuery"</ion-input>
16       </ion-item>
17     </ion-list>
18     <button ion-button (click)="doSearch()">검색해보기</button>
19   </div>
20   <div>
21     <ion-list>
22       <a ion-item *ngFor="let result of searchResult" href="{{_
23         result.link }}">
24         <strong style="display:block;">{{ result.title }}</strong>
25         <span style="display:block">{{ result.description }}</span>
26       </a>
27     </ion-list>
28   </div>
29 </ion-content>
```

restful API

- src/pages/home/home.ts

```
1  import { Component } from '@angular/core';
2  import { NavController } from 'ionic-angular';
3  import { HttpClient, HttpHeaders } from '@angular/common/http';
4
5  @Component({
6    selector: 'page-home',
7    templateUrl: 'home.html'
8  })
9
10 export class HomePage {
11
12   searchQuery:string;
13   searchResult;
14
15   constructor(public navCtrl: NavController, private
16   http:HttpClient) {
17
18
19   doSearch() {
20     let url = 'http://atstudio.co.kr/naver_search_test.php?
21     query=' + this.searchQuery;
22
23     this.http.get(url).subscribe((res:any) => {
24       this.searchResult = res.items;
25     }, (err) => {
26       console.error('오류 발생');
27     });
28   }
29 }
```

restful API

- POST 사용해보기
 - URL에 요청 데이터(파라미터)를 실어서 보내지 않고, HTTP 프로토콜의 요청 바디 내부에 데이터를 포함하여 전송.
 - 서버로 보내려는 데이터가 숨겨져 있으므로 GET보다 효율적.
 - ionic start rest2 blank

로그인

아이디	아이디 입력
비밀번호	비밀번호 입력

[로그인 하기](#)

restful API

- POST : 시나리오
 - 가상의 서버에 로그인 계정이 존재하며, 이는 ID와 비밀번호가 각각 test / happy 로 존재함을 가정.
 - test 계정 외에 접근 불허.

restful API

- src/app/app.module.ts

```
1 | import { HttpClientModule } from '@angular/common/http';
```

```
16    imports: [
17      BrowserModule,
18      HttpClientModule,
19      IonicModule.forRoot(MyApp)
20    ],
```

restful API

- src/pages/home/home.html

```
1  <ion-header>
2    <ion-navbar>
3      <ion-title>
4        로그인
5      </ion-title>
6    </ion-navbar>
7  </ion-header>
8
9  <ion-content padding>
10 <ion-list>
11   <ion-item>
12     <ion-label fixed>아이디</ion-label>
13     <ion-input type="text" placeholder="아이디 입력" [(ngModel)] = "uid"></ion-input>
14   </ion-item>
15   <ion-item>
16     <ion-label fixed>비밀번호</ion-label>
17     <ion-input type="password" placeholder="비밀번호 입력" [(ngModel)] = "upw"></ion-input>
18   </ion-item>
19 </ion-list>
20 <button ion-button (click)="login()">로그인 하기</button>
21 <p>{{ responseText }}</p>
22 </ion-content>
```

restful API

- src/pages/home/home.ts

```
1 import { Component } from '@angular/core';
2 import { NavController } from 'ionic-angular';
3 import { HttpClient, HttpHeaders } from '@angular/common/http';
4
5 @Component({
6   selector: 'page-home',
7   templateUrl: 'home.html'
8 })
9 export class HomePage {
10   uid:string;
11   upw:string;
12   responseText;
13
14   constructor(public navCtrl: NavController, private
15   http:HttpClient) {
16
17
18   login() {
19     let url = 'http://atstudio.co.kr/ionic_post_test.php';
20     let headers = new HttpHeaders({'Content-Type':
21       'application/x-www-form-urlencoded'});
22     let body = 'uid=' + this.uid + '&upw=' + this.upw;
23
24     this.http.post(url, body, {headers: headers}).subscribe(
25       (res:any) => [
26         this.responseText = res.message;
27       ], (err) => {
28         console.error('오류발생');
29       });
30   }
31 }
```

오늘은 여기까지~

See you next day!