Programming II

가성비 최고의 웹/하이브리드 앱 개발

Foreword
머릿말

Ever since smartphones came out in the world, the world has faced a revolution. People started making transactions, video chat with their family and colleagues, play game with their phone! Moreover, the development team of the features is independent to the phone companies. The phone technology enabled software developers to make an application that can be used in any device sharing the same Operating System (OS).

스마트폰이 세상에 나오고 나서 부터 세상은 혁신을 맞이합니다. 사람들은 핸드폰으로 은행업무, 친구 또는 직장 동료와의 화상통화, 게임을 합니다! 심지어 그런 기술의 개발팀은 핸드폰 회와 독립적입니다. 핸드폰 기술의 발전은 소프트웨어 개발자들로 하여금 동일 운영체제(OS)를 쓰는 어떠한 기기에서도 구동될 수 있는 프로그램을 만들수 있게 하였습니다.

This book is published in june 24, 2019. At this time, the two promising phone OSs are android and iOS. Each has their own Integrated Development Environment (IDE), and they both can access the internet. Some have made new IDEs for both OSs or both OSs and the internet browsers so app developers can code once and use it in multiple platforms.

이 책은 2019년 6월 24일에 쓰여지기 시작했습니다. 이때에는, 안드로이드와 iOS의 OS가 널리 쓰이고 있는 상황입니다. 각 플랫폼은 그들만의 코딩환경이 있고 그들 둘 다 인터넷 접속이 가능합니다. 그리고 두 OS에서 통용되는 어플을 만들수 있는 코딩환경, 더 나아가서는 두 OS와 인터넷 브라우저에서 모두 열릴 수 있는 코딩환경이 개발된 상황입니다. 코드 한번에 여러 플랫폼에서 쓰이는 상황이죠.

Actually, there are many platforms that can run on both Android, iOS, and even multiple other platforms. It is quite difficult to say which one is better than which; however, this book can briefly go over which platform is right for which case. Now, remember this book is published in jun 24, 2019, so the following information only makes sense in that moment, which will most likely be the past to the readers of this book. The author encourages readers to find the right platform for their own apps!
사실 안드로이드, iOS 기타 등등을 모두 아우르는 플랫폼들은 많이 있습니다. 특정한 플랫폼을 최고라고 말하기는 어렵습니다만, 특정 어플에게는 특정 플랫폼이 좋다라고는 설명드릴 수 있습니다. 다시 한번 이 책이 출간된 날짜를 기억해주세요. 앞으로 말씀드릴 플랫폼에 대한 정보는 미래의 독자들에게 사실이 아닐 수 있습니다. 따라서 꼭 직접 어떤 플랫폼이 본인이 구상하는 앱을 위한 최고의 플랫폼인지 확인해주시기 바랍니다.

All that being said, the author can think of three ways: performance, cost, and accessibility. The best platform in performance is native platforms such as android studio and xcode. For this, you need to code n times for n OSs. For instance, if you want to run your application on Android, iOS, and internet browser, you will need to code three times from scratch. This will give you the best performance. However, You'll need three different development teams to build three different apps. This leads to consider the cost, because the cost for building your app,

maintenance of your app, and upgrading your app will be the highest. This is why hybrid platforms exist. They are either as fast or slower than native platforms, but using them allows people to reuse their code for other platforms.

This book utilizes one of the IDEs called Ionic, and the version is 4.
이 책은 아이오닉이라는 코딩환경을 이용할 것이고 버전은 4입니다.

Ionic 4 is a web app tool, which makes a browser screen when it is opened in browsers, android and iOS. However, when used in android and iOS, it hides the address bar so users won't feel it is in browser environment.

아이오닉 4는 웹앱 툴입니다. 따라서 인터넷 브라우저, 안드로이드, iOS에서 열릴때 모두, 인터넷 브라우저를 엽니다. 하지만, 안드로이드와 iOS에서 열릴 때에는 주소창을 가려서 사용자가 인터넷 환경인지 인지하지 못하게 합니다.

앱만들기 ionic 4

I. 입문
   A. Value-Containing Component
      1. Ion-checkbox
      2. Ion-datetime & Ion-picker
      3. ion-radio
      4. Ion-range
      5. Ion-segment
      6. Ion-select
      7. Ion-toggle
   B. Function-Containing Component
      1. Ion-action-sheet
      2. Ion-alert
      3. Ion-fab
      4. Ion-popover
   C. BitCoin$ Project
      1. I-C-1 http로 코인값 가져오기 (link)
      2. I-C-2 다다다다 처리(link)
      3. I-C-3 버튼 디자인 (link)
      4. I-C-4 Array 처리 (link)
      5. I-C-5 그래프 입문 (link)
      6. I-C-6 그래프 처리 (link)

II. 도약
   A. Navigation
      1. Ion-route & Ion-router
      2. Ion-tabs
   B. Reactive Component
      1. Ion-input, Ion-item
      2. Ion-list, Ion-card
      3. Ion-searchbar
   C. Firebase
      1. Firebase 투어 및 Authentication과 Database의 차이
      2. Authentication by Email
      3. Real Time Database
      4. Firestore
   D. 변호사 플랫폼 프로젝트 입문
      1. UI & Storyboard

III. MVP
   A. Components For better User Experience
      1. Ion-infinite-scroll & Ion-refresher

2. Ion-loading & ion-spinner & Ion-toast
3. Ion-slides

B. Firebase 심화
1. Storage
2. Functions

C. 변호사 플랫폼 프로젝트 완성
1. *compile과 translating의 차이
2. Web
3. Firebase hosting
4. Android
5. IOS

IV. 창업

A. Firebase 심화단계
1. Security Firebase Functions
2. Authentication by Phone
3. Promise.all()

B. 이론
1.
2.

C. Note App Project with Google Map and Camera

D. Asthetics
1. Ion-badge
2. Ion-ripple-effect
3. Ion-chip
4. Ion-grid
5. Ion-icon (xd)
6. Ion-avatar
7. Ion-img & ion-thumbnail

V. Database : Why not utilizing firebase

A. Database
1. KEY
2. Update
3. Rules

B. Auth
1. Signup & Signin
2. Email
3. Deeplink
4. Phone

C. Schedule App Project

D. Function
1. Basic Function Operation

        2. onCreate, onUpdate, onDelete
    E. Store
    F. Hosting

VI. More
    A. generations
        1. pageName.service.ts
        2. pageName.guard.ts
        3. pageName.component.ts
        4. providers
    B. Type
    C. Asynchronous Operations

1. Component

In this chapter, we'll review each component in the link (https://ionicframework.com/docs). The author thinks it is the best way to motivate readers by guiding them realizing things. what they have done for the real world when learning challenging things such as making an app.

이 챕터는 아이오닉 프레임워크에서 제공하는 컴포넌트를 위 링크에 있는 컴포넌트들에 대하여 리뷰할 것입니다. 저자는 앱만드는것과 같이 어려운 과정을 배우는데에 있어 최고의 동기부여 방법은 실제로 구현한 것을 바로 확인하는것이라 생각합니다.

At first, this chapter will cover the value-containing components; followed by function-containing components and navigation. By these three sub-chapters, readers will be able to make fundamental apps. The next is some tools which the author calls "for better user experience". It makes the user experience more fruitful. In many times, apps take users' input in the form, so the next sub-chapter introduces grouping tools and input. Finally, design-oriented features will be introduced and it finalizes the component chapter.

처음에 이 챕터는 값을 가지는 컴포넌트, 함수를 가지는 컴포넌트, 네비게이션 컴포넌트에 대하여 배울것입니다. 이 세가지만으로도 근본적인 어플은 만들 수 있습니다. 다음으로는 "UX향상을 위한" 부분을 배울것입니다. 이것으로 인해 어플을 더욱 풍성하게 만들 수 있습니다. 많은 어플이 사용자의 인풋을 받기 때문에 다음 레슨은 인풋과 동시에 컴포넌트를 그룹하는 방법을 배울것입니다. 마지막으로 디자인 성향이 강한 컴포넌트들을 배우며 컴포넌트 챕터는 막을 내릴것입니다.

The readers will have two project

a. Value-Containing

Ion-checkbox

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
```

```
</ion-header>

<ion-content>
 <ion-checkbox [(ngModel)]="firstCheckbox" ></ion-checkbox>
 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>
</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
 selector: 'app-home',
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private firstCheckbox: boolean;

 constructor() {}



}
```

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>
```

```
 <ion-item>
   <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
   <ion-checkbox slot="start" [(ngModel)]="firstCheckbox" ></ion-checkbox>
 </ion-item>

 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>
</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
 selector: 'app-home',
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private firstCheckbox: boolean;

 constructor() {}

}
```

Ion-datetime
Ion-picker

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
```

```
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>


  <ion-item>
    <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
    <ion-checkbox slot="end" [(ngModel)]="firstCheckbox" ></ion-checkbox>
  </ion-item>

  <ion-grid>
    <ion-row>
      <h1 *ngIf="firstCheckbox" >체크했다!</h1>
    </ion-row>
  </ion-grid>

  <ion-item>
    <ion-label>Date</ion-label>
    <ion-datetime
      display-format="MMM DD, YYYY"
      [(ngModel)]="firstDate"
    ></ion-datetime>
  </ion-item>

  <ion-grid>
    <ion-row>
      <ion-button (click)="customTime()">커스텀시간가져오기</ion-button>
    </ion-row>
  </ion-grid>

  <ion-grid>
    <ion-row>
      <h5 *ngIf="firstCheckbox" >{{firstDate}}</h5>
      <h5 >{{myString}}</h5>
    </ion-row>
  </ion-grid>
```

```
</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
 selector: 'app-home',
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private firstCheckbox: boolean;
 private firstDate: string;
 private myString = '변신 전!';

 constructor() {}

 customTime() {
   const now = new Date(this.firstDate);
   const hours = now.getHours();
   this.myString = hours.toString();
 }

}
```

Ion-radio

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>
```

```
<ion-content>


 <ion-item>
   <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
   <ion-checkbox slot="end" [(ngModel)]="firstCheckbox" ></ion-checkbox>
 </ion-item>

 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>

 <ion-item>
   <ion-label>Date</ion-label>
   <ion-datetime
     display-format="MMM DD, YYYY"
     [(ngModel)]="firstDate"
   ></ion-datetime>
 </ion-item>

 <ion-grid>
   <ion-row>
     <ion-button (click)="customTime()">커스텀시간가져오기</ion-button>
   </ion-row>
 </ion-grid>

 <ion-grid>
   <ion-row>
     <h5 *ngIf="firstCheckbox" >{{firstDate}}</h5>
     <h5 >{{myString}}</h5>
   </ion-row>
 </ion-grid>

 <ion-radio-group [(ngModel)]="firstRadio" >
   <ion-list-header>
     <ion-label>라디오예제</ion-label>
   </ion-list-header>
```

```
    <ion-item>
      <ion-label>1번</ion-label>
      <ion-radio slot="start" value="1"></ion-radio>
    </ion-item>
    <ion-item>
      <ion-label>2번</ion-label>
      <ion-radio slot="start" value="2"></ion-radio>
    </ion-item>
  </ion-radio-group>

  <ion-grid>
    <ion-row>
      <h5 *ngIf="firstRadio === '1'" >라디오 1번!</h5>
      <h5 *ngIf="firstRadio === '2'" >라디오 2번!</h5>
    </ion-row>
  </ion-grid>

</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  private firstCheckbox: boolean;
  private firstDate: string;
  private myString = '변신 전!';
  private firstRadio;

  constructor() {}
```

```
 customTime() {
   const now = new Date(this.firstDate);
   const hours = now.getHours();
   this.myString = hours.toString();
 }

}
```

Ion-toggle

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>


 <ion-item>
   <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
   <ion-checkbox slot="end" [(ngModel)]="firstCheckbox" ></ion-checkbox>
 </ion-item>

 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>

 <ion-item>
   <ion-label>Date</ion-label>
   <ion-datetime
     display-format="MMM DD, YYYY"
     [(ngModel)]="firstDate"
   ></ion-datetime>
 </ion-item>
```

```
<ion-grid>
  <ion-row>
    <ion-button (click)="customTime()">커스텀시간가져오기</ion-button>
  </ion-row>
</ion-grid>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstCheckbox" >{{firstDate}}</h5>
    <h5 >{{myString}}</h5>
  </ion-row>
</ion-grid>

<ion-radio-group [(ngModel)]="firstRadio" >
  <ion-list-header>
    <ion-label>라디오예제</ion-label>
  </ion-list-header>

  <ion-item>
    <ion-label>1번</ion-label>
    <ion-radio slot="start" value="1"></ion-radio>
  </ion-item>

  <ion-item>
    <ion-label>2번</ion-label>
    <ion-radio slot="start" value="2"></ion-radio>
  </ion-item>
</ion-radio-group>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstRadio === '1'" >라디오 1번!</h5>
    <h5 *ngIf="firstRadio === '2'" >라디오 2번!</h5>
  </ion-row>
</ion-grid>


<ion-item>
```

```
    <ion-label>토글 예제</ion-label>
    <ion-toggle [(ngModel)]="firstToggle"></ion-toggle>
  </ion-item>

  <ion-grid>
    <ion-row>
      <h5 *ngIf="firstToggle" >토글 선택했다!</h5>
    </ion-row>
  </ion-grid>

</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  private firstCheckbox: boolean;
  private firstDate: string;
  private myString = '변신 전!';
  private firstRadio;
  private firstToggle: boolean;

  constructor() {}

  customTime() {
    const now = new Date(this.firstDate);
    const hours = now.getHours();
    this.myString = hours.toString();
  }

}
```

Ion-range

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>



 <ion-item>
   <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
   <ion-checkbox slot="end" [(ngModel)]="firstCheckbox" ></ion-checkbox>
 </ion-item>

 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>

 <ion-item>
   <ion-label>Date</ion-label>
   <ion-datetime
     display-format="MMM DD, YYYY"
     [(ngModel)]="firstDate"
   ></ion-datetime>
 </ion-item>

 <ion-grid>
   <ion-row>
     <ion-button (click)="customTime()">커스텀시간가져오기</ion-button>
   </ion-row>
 </ion-grid>
```

```
<ion-grid>
  <ion-row>
    <h5 *ngIf="firstCheckbox" >{{firstDate}}</h5>
    <h5 >{{myString}}</h5>
  </ion-row>
</ion-grid>

<ion-radio-group [(ngModel)]="firstRadio" >
  <ion-list-header>
    <ion-label>라디오예제</ion-label>
  </ion-list-header>

  <ion-item>
    <ion-label>1번</ion-label>
    <ion-radio slot="start" value="1"></ion-radio>
  </ion-item>

  <ion-item>
    <ion-label>2번</ion-label>
    <ion-radio slot="start" value="2"></ion-radio>
  </ion-item>
</ion-radio-group>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstRadio === '1'" >라디오 1번!</h5>
    <h5 *ngIf="firstRadio === '2'" >라디오 2번!</h5>
  </ion-row>
</ion-grid>



<ion-item>
  <ion-label>토글 예제</ion-label>
  <ion-toggle [(ngModel)]="firstToggle"></ion-toggle>
</ion-item>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstToggle" >토글 선택했다!</h5>
```

```
    </ion-row>
  </ion-grid>

  <ion-item>
    <ion-label>말벌인가?</ion-label>
    <ion-range min="-200" max="200" [(ngModel)]="firstRange" >
      <ion-label slot="start">매우 불확실</ion-label>
      <ion-label slot="end">매우 확실</ion-label>
    </ion-range>
  </ion-item>

  <ion-grid>
    <ion-row>
      <h5 >{{firstRange}}</h5>
    </ion-row>
  </ion-grid>

</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  private firstCheckbox: boolean;
  private firstDate: string;
  private myString = '변신 전!';
  private firstRadio;
  private firstToggle: boolean;
  private firstRange;

  constructor() {}
```

```
 customTime() {
   const now = new Date(this.firstDate);
   const hours = now.getHours();
   this.myString = hours.toString();
 }

}
```

Ion-segment

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>


 <ion-item>
   <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
   <ion-checkbox slot="end" [(ngModel)]="firstCheckbox" ></ion-checkbox>
 </ion-item>

 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>

 <ion-item>
   <ion-label>Date</ion-label>
   <ion-datetime
     display-format="MMM DD, YYYY"
     [(ngModel)]="firstDate"
   ></ion-datetime>
```

```
</ion-item>

<ion-grid>
  <ion-row>
    <ion-button (click)="customTime()">커스텀시간가져오기</ion-button>
  </ion-row>
</ion-grid>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstCheckbox" >{{firstDate}}</h5>
    <h5 >{{myString}}</h5>
  </ion-row>
</ion-grid>

<ion-radio-group [(ngModel)]="firstRadio" >
  <ion-list-header>
    <ion-label>라디오예제</ion-label>
  </ion-list-header>

  <ion-item>
    <ion-label>1번</ion-label>
    <ion-radio slot="start" value="1"></ion-radio>
  </ion-item>

  <ion-item>
    <ion-label>2번</ion-label>
    <ion-radio slot="start" value="2"></ion-radio>
  </ion-item>
</ion-radio-group>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstRadio === '1'" >라디오 1번!</h5>
    <h5 *ngIf="firstRadio === '2'" >라디오 2번!</h5>
  </ion-row>
</ion-grid>
```

```
<ion-item>
  <ion-label>토글 예제</ion-label>
  <ion-toggle [(ngModel)]="firstToggle"></ion-toggle>
</ion-item>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstToggle" >토글 선택했다!</h5>
  </ion-row>
</ion-grid>

<ion-item>
  <ion-label>말벌인가?</ion-label>
  <ion-range min="-200" max="200" [(ngModel)]="firstRange" >
    <ion-label slot="start">매우 불확실</ion-label>
    <ion-label slot="end">매우 확실</ion-label>
  </ion-range>
</ion-item>

<ion-grid>
  <ion-row>
    <h5 >{{firstRange}}</h5>
  </ion-row>
</ion-grid>

<ion-segment (ionChange)="segmentChanged($event)" [(ngModel)]="segOption" >
  <ion-segment-button value="option1">
    <ion-label>친구</ion-label>
  </ion-segment-button>
  <ion-segment-button value="option2">
    <ion-label>여자친구</ion-label>
  </ion-segment-button>
</ion-segment>

<ion-grid>
  <ion-row>
    <h1>s</h1>
    <h5 >{{firstSegMsg}}</h5>
  </ion-row>
```

```
 </ion-grid>

</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
 selector: 'app-home',
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private firstCheckbox: boolean;
 private firstDate: string;
 private myString = '변신 전!';
 private firstRadio;
 private firstToggle: boolean;
 private firstRange;
 private firstSegMsg;
 private segOption;

 constructor() {}

 customTime() {
   const now = new Date(this.firstDate);
   const hours = now.getHours();
   this.myString = hours.toString();
 }

 segmentChanged($event) {
   if ( this.segOption === 'option1' ) {
     this.firstSegMsg = '친구 선택했구나~';
   } else if ( this.segOption === 'option2' ) {
     this.firstSegMsg = '여자친구친구 선택했구나~';
   }
 }
```

```
}
```

Ion-select

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>


 <ion-item>
   <ion-label>처음 해보는 체크박스 ㅎㅎㅎ</ion-label>
   <ion-checkbox slot="end" [(ngModel)]="firstCheckbox" ></ion-checkbox>
 </ion-item>

 <ion-grid>
   <ion-row>
     <h1 *ngIf="firstCheckbox" >체크했다!</h1>
   </ion-row>
 </ion-grid>

 <ion-item>
   <ion-label>Date</ion-label>
   <ion-datetime
     display-format="MMM DD, YYYY"
     [(ngModel)]="firstDate"
   ></ion-datetime>
 </ion-item>

 <ion-grid>
   <ion-row>
     <ion-button (click)="customTime()">커스텀시간가져오기</ion-button>
   </ion-row>
```

```
</ion-grid>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstCheckbox" >{{firstDate}}</h5>
    <h5 >{{myString}}</h5>
  </ion-row>
</ion-grid>

<ion-radio-group [(ngModel)]="firstRadio" >
  <ion-list-header>
    <ion-label>라디오예제</ion-label>
  </ion-list-header>

  <ion-item>
    <ion-label>1번</ion-label>
    <ion-radio slot="start" value="1"></ion-radio>
  </ion-item>

  <ion-item>
    <ion-label>2번</ion-label>
    <ion-radio slot="start" value="2"></ion-radio>
  </ion-item>
</ion-radio-group>

<ion-grid>
  <ion-row>
    <h5 *ngIf="firstRadio === '1'" >라디오 1번!</h5>
    <h5 *ngIf="firstRadio === '2'" >라디오 2번!</h5>
  </ion-row>
</ion-grid>


<ion-item>
  <ion-label>토글 예제</ion-label>
  <ion-toggle [(ngModel)]="firstToggle"></ion-toggle>
</ion-item>

<ion-grid>
```

```
  <ion-row>
    <h5 *ngIf="firstToggle" >토글 선택했다!</h5>
  </ion-row>
</ion-grid>

<ion-item>
  <ion-label>말벌인가?</ion-label>
  <ion-range min="-200" max="200" [(ngModel)]="firstRange" >
    <ion-label slot="start">매우 불확실</ion-label>
    <ion-label slot="end">매우 확실</ion-label>
  </ion-range>
</ion-item>

<ion-grid>
  <ion-row>
    <h5 >{{firstRange}}</h5>
  </ion-row>
</ion-grid>

<ion-segment (ionChange)="segmentChanged($event)" [(ngModel)]="segOption" >
  <ion-segment-button value="option1">
    <ion-label>친구</ion-label>
  </ion-segment-button>
  <ion-segment-button value="option2">
    <ion-label>여자친구</ion-label>
  </ion-segment-button>
</ion-segment>

<ion-grid>
  <ion-row>
    <h5 >{{firstSegMsg}}</h5>
  </ion-row>
</ion-grid>
 <ion-grid>
  <ion-row>
    <h5>성별은?</h5>
    <h5 >{{firstGMsg}}</h5>
  </ion-row>
</ion-grid>
```

```
<ion-grid>
    <ion-row>
      <h5>당신의 대륙은 : </h5>
      <h5 > {{firstCMsg}}</h5>
    </ion-row>
</ion-grid>

<ion-grid>
    <ion-row>
      <h5>당신의 점심은 : </h5>
      <h5 > {{firstLMsg}}</h5>
    </ion-row>
</ion-grid>

<ion-list>
  <ion-list-header>select 예제</ion-list-header>

  <ion-item>
    <ion-label>성별</ion-label>
    <ion-select placeholder="Select One" interface="alert"
(ionChange)="genderChange($event)" [(ngModel)]="gender">
      <ion-select-option value="f">여자</ion-select-option>
      <ion-select-option value="m">남자</ion-select-option>
    </ion-select>
  </ion-item>
   <ion-item>
    <ion-label>대륙</ion-label>
    <ion-select value="brown" interface="popover" okText="Okay" cancelText="Dismiss"
(ionChange)="continentChange($event)" [(ngModel)]="continent">
      <ion-select-option value="asia">아시아</ion-select-option>
      <ion-select-option value="europe">유럽</ion-select-option>
    </ion-select>
  </ion-item>
   <ion-item>
    <ion-label>점심 뭐먹지</ion-label>
    <ion-select value="brown" interface="action-sheet" okText="Okay"
cancelText="Dismiss" (ionChange)="lunchChange($event)" [(ngModel)]="lunch">
      <ion-select-option value="china">중국집</ion-select-option>
```

```
        <ion-select-option value="japan">일식집</ion-select-option>
      </ion-select>
    </ion-item>
  </ion-list>


</ion-content>
```

In typescript

```
import { Component } from '@angular/core';

@Component({
 selector: 'app-home',
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private firstCheckbox: boolean;
 private firstDate: string;
 private myString = '변신 전!';
 private firstRadio;
 private firstToggle: boolean;
 private firstRange;
 private firstSegMsg;
 private segOption;
 private gender;
 private firstGMsg;
 private continent;
 private firstCMsg;
 private lunch;
 private firstLMsg;

 constructor() {}

 customTime() {
   const now = new Date(this.firstDate);
   const hours = now.getHours();
   this.myString = hours.toString();
```

```
  }

  segmentChanged($event) {
    if ( this.segOption === 'option1' ) {
      this.firstSegMsg = '친구 선택했구나~';
    } else if ( this.segOption === 'option2' ) {
      this.firstSegMsg = '여자친구친구 선택했구나~';
    }
  }

  genderChange($event) {
    if (this.gender === 'f') {
      this.firstGMsg = '여자';
    } else if (this.gender === 'm') {
      this.firstGMsg = '남자';
    }
  }

  continentChange($event) {
    if (this.continent === 'asia') {
      this.firstCMsg = '아시아';
    } else if (this.continent === 'europe') {
      this.firstCMsg = '유럽';
    }
  }

  lunchChange($event) {
    if (this.lunch === 'china') {
      this.firstLMsg = '중국집';
    } else if (this.lunch === 'japan') {
      this.firstLMsg = '일식집';
    }
  }

}
```

b. Function-Containing

Ion-action-sheet

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Function Containing
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>
 <div>
   <ion-grid>
     <ion-row>
       <ion-col>
         <ion-row>
           <ion-button (click)="presentActionSheet()" >Action Sheet</ion-button>
         </ion-row>
       </ion-col>
       <ion-col>
         <ion-row justify-content-end >
           <h5 >{{actionSheetMsg}}</h5>
         </ion-row>
       </ion-col>
     </ion-row>
   </ion-grid>
 </div>
</ion-content>
```

In ts

```
import { Component } from '@angular/core';
import { ActionSheetController } from '@ionic/angular';

@Component({
 selector: 'app-home',
```

```
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private actionSheetMsg: number;

 constructor(
   private actionSheetCtrl: ActionSheetController,
 ) {}

 async presentActionSheet() {
   const actionSheet = await this.actionSheetCtrl.create({
     header: '액션시트를 통해 값을 지정해보자',
     buttons: [{
       text: '0',
       icon: 'trash',
       handler: () => {
         this.actionSheetMsg = 0;
       }
     }, {
       text: '1',
       icon: 'share',
       handler: () => {
         this.actionSheetMsg = 1;
       }
     }, {
       text: '2',
       icon: 'arrow-dropright-circle',
       handler: () => {
         this.actionSheetMsg = 2;
       }
     }, {
       text: '3',
       icon: 'heart',
       handler: () => {
         this.actionSheetMsg = 3;
       }
     }, {
```

```
        text: 'Cancel',
        icon: 'close',
        role: 'cancel',
        // handler: () => {
        //   console.log('Cancel clicked');
        // }
      }]
    });
    await actionSheet.present();
  }

}
```

Ion-alert

In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Function Containing
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>
 <div>
   <ion-grid>
     <ion-row>
       <ion-col>
         <ion-row>
           <ion-button (click)="presentActionSheet()" >Action Sheet</ion-button>
         </ion-row>
       </ion-col>
       <ion-col>
         <ion-row justify-content-end >
           <h5 >{{actionSheetMsg}}</h5>
         </ion-row>
       </ion-col>
     </ion-row>
```

```
    </ion-grid>
  </div>
</ion-content>
```

In ts

```
import { Component } from '@angular/core';
import { ActionSheetController, AlertController } from '@ionic/angular';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  private actionSheetMsg: number;

  constructor(
    private actionSheetCtrl: ActionSheetController,
    private alertCtrl: AlertController,
  ) {}

  async presentActionSheet() {
    const actionSheet = await this.actionSheetCtrl.create({
      header: '액션시트를 통해 값을 지정해보자',
      buttons: [{
        text: 'alert 예제 확인',
        icon: 'trash',
        handler: async () => {
          const alert = await this.alertCtrl.create({
            header: 'Alert 눌렀냐?',
            subHeader: 'Alert 예제 공부하나부네~',
            message: '경고메세지야!',
            buttons: [
              {
                text: '0 나와라!',
                handler: () => {
                  this.actionSheetMsg = 0;
```

```
            }
          },
          {
            text: '취소',
            role: 'cancel',
          }
        ]
      });
      await alert.present();
    }
  }, {
    text: '1',
    icon: 'share',
    handler: () => {
      this.actionSheetMsg = 1;
    }
  }, {
    text: '2',
    icon: 'arrow-dropright-circle',
    handler: () => {
      this.actionSheetMsg = 2;
    }
  }, {
    text: '3',
    icon: 'heart',
    handler: () => {
      this.actionSheetMsg = 3;
    }
  }, {
    text: 'Cancel',
    icon: 'close',
    role: 'cancel',
    // handler: () => {
    //   console.log('Cancel clicked');
    // }
  }]
 });
 await actionSheet.present();
}
```

```
}
```

Ion-button 수없이 많이 했으므로 생략
In html

In ts

Ion-fab
In html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Function Containing
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>
 <div>
   <ion-grid>
     <ion-row>
       <ion-col>
         <ion-row>
           <ion-button (click)="presentActionSheet()" >Action Sheet</ion-button>
         </ion-row>
       </ion-col>
       <ion-col>
         <ion-row justify-content-end >
           <h5 >{{actionSheetMsg}}</h5>
         </ion-row>
       </ion-col>
     </ion-row>
   </ion-grid>
 </div>
```

```
 <div>

   <ion-fab vertical="center" horizontal="center" slot="fixed">
     <ion-fab-button>
       <ion-icon name="share"></ion-icon>
     </ion-fab-button>
     <ion-fab-list side="top">
       <ion-fab-button (click)="fab('logo-vimeo')" ><ion-icon
name="logo-vimeo"></ion-icon></ion-fab-button>
     </ion-fab-list>
     <ion-fab-list side="bottom">
       <ion-fab-button (click)="fab('logo-facebook')" ><ion-icon
name="logo-facebook"></ion-icon></ion-fab-button>
     </ion-fab-list>
     <ion-fab-list side="start">
       <ion-fab-button (click)="fab('logo-instagram')" ><ion-icon
name="logo-instagram"></ion-icon></ion-fab-button>
     </ion-fab-list>
     <ion-fab-list side="end">
       <ion-fab-button (click)="fab('logo-twitter')" ><ion-icon
name="logo-twitter"></ion-icon></ion-fab-button>
     </ion-fab-list>
   </ion-fab>

 </div>

 <div>
   <ion-grid>
     <ion-row justify-content-center >
       <h5>{{fabMsg}} </h5>
     </ion-row>
   </ion-grid>
 </div>
```

```
</ion-content>
```

In ts

```
import { Component } from '@angular/core';
import { ActionSheetController, AlertController } from '@ionic/angular';

@Component({
 selector: 'app-home',
 templateUrl: 'home.page.html',
 styleUrls: ['home.page.scss'],
})
export class HomePage {

 private actionSheetMsg: number;
 private fabMsg: string;

 constructor(
   private actionSheetCtrl: ActionSheetController,
   private alertCtrl: AlertController,
 ) {}

 async presentActionSheet() {
   const actionSheet = await this.actionSheetCtrl.create({
     header: '액션시트를 통해 값을 지정해보자',
     buttons: [{
       text: 'alert 에제 확인',
       icon: 'trash',
       handler: async () => {
         const alert = await this.alertCtrl.create({
           header: 'Alert 눌렀냐?',
           subHeader: 'Alert 에제 공부하나부네~',
           message: '경고메세지야!',
           buttons: [
             {
               text: '0 나와라!',
               handler: () => {
                 this.actionSheetMsg = 0;
               }
```

```
            },
            {
              text: '취소',
              role: 'cancel',
            }
          ]
        });
        await alert.present();
      }
    }, {
      text: '1',
      icon: 'share',
      handler: () => {
        this.actionSheetMsg = 1;
      }
    }, {
      text: '2',
      icon: 'arrow-dropright-circle',
      handler: () => {
        this.actionSheetMsg = 2;
      }
    }, {
      text: '3',
      icon: 'heart',
      handler: () => {
        this.actionSheetMsg = 3;
      }
    }, {
      text: 'Cancel',
      icon: 'close',
      role: 'cancel',
      // handler: () => {
      //   console.log('Cancel clicked');
      // }
    }]
  });
  await actionSheet.present();
}
```

```
 fab(btnString: string) {
   this.fabMsg = btnString;
 }

}
```

Ion-popover
Open terminal and set the command directory to your project folder. Then put in the following:
터미널을 여시고 폴더를 프로젝트 폴더로 지정하세요. 그런 다음, 아래 코드를 입력하세요.

Ionic g component pop

This will create a folder called pop, and you will be able to check it in your VS Code environment. We are going to use this pop component that we just created in home.page.ts.
위 코드를 입력하시면, “pop”이라는 이름의 새로운 폴더가 생길것입니다. VS Code에서 폴더가 잘 만들어졌는지 확인하시기 바랍니다. 방금 생성한 pop 컴포넌트를 home.page.ts에서 사용할 것입니다. 아래 코드를 입력해주시기 바랍니다.

In pop.component.html

```
<ion-list>
 <ion-list-header >
     <ion-label >Notifications</ion-label>
   </ion-list-header>

   <ion-item (click)="messages(25)">
     <ion-icon name="mail" color="primary"></ion-icon>
     <ion-label>New Offer 25% OFF</ion-label>
   </ion-item>

   <ion-item (click)="messages(15)">
     <ion-icon name="mail-open" color="primary"></ion-icon>
     <ion-label>New Offer 15% OFF by month!</ion-label>
   </ion-item>
</ion-list>
```

In pop.component.ts

```
import { Component, OnInit } from '@angular/core';
```

```
import { PopoverController } from '@ionic/angular';

@Component({
 selector: 'app-pop',
 templateUrl: './pop.component.html',
 styleUrls: ['./pop.component.scss'],
})
export class PopComponent implements OnInit {

 constructor(
   private popover: PopoverController
 ) { }

 ngOnInit() {}

 messages(val: number) {
   console.log(`${val} can be saved!`);
   this.popover.dismiss();
 }

}
```

In home.page.html

```
<ion-header>
 <ion-toolbar>
   <ion-title>
     Function Containing
   </ion-title>
 </ion-toolbar>
</ion-header>

<ion-content>
 <div>
   <ion-grid>
     <ion-row>
       <ion-col>
         <ion-row>
           <ion-button (click)="presentActionSheet()" >Action Sheet</ion-button>
```

```
        </ion-row>
      </ion-col>
      <ion-col>
        <ion-row justify-content-end >
          <h5 >{{actionSheetMsg}}</h5>
        </ion-row>
      </ion-col>
    </ion-row>
  </ion-grid>
</div>




<div>

  <ion-fab vertical="center" horizontal="center" slot="fixed">
    <ion-fab-button>
      <ion-icon name="share"></ion-icon>
    </ion-fab-button>
    <ion-fab-list side="top">
      <ion-fab-button (click)="fab('logo-vimeo')" ><ion-icon
name="logo-vimeo"></ion-icon></ion-fab-button>
    </ion-fab-list>
    <ion-fab-list side="bottom">
      <ion-fab-button (click)="fab('logo-facebook')" ><ion-icon
name="logo-facebook"></ion-icon></ion-fab-button>
    </ion-fab-list>
    <ion-fab-list side="start">
      <ion-fab-button (click)="fab('logo-instagram')" ><ion-icon
name="logo-instagram"></ion-icon></ion-fab-button>
    </ion-fab-list>
    <ion-fab-list side="end">
      <ion-fab-button (click)="fab('logo-twitter')" ><ion-icon
name="logo-twitter"></ion-icon></ion-fab-button>
    </ion-fab-list>
  </ion-fab>
```

```
 </div>

 <div>
   <ion-grid>
     <ion-row justify-content-center >
       <h5>{{fabMsg}} </h5>
     </ion-row>
   </ion-grid>
 </div>

 <div>
   <ion-grid>
     <ion-row>
       <ion-button (click)="presentPopover(ev)">
         popoverp
       </ion-button>
     </ion-row>
   </ion-grid>
 </div>

</ion-content>
```

In home.page.ts

```
import { Component } from '@angular/core';
import { ActionSheetController, AlertController, PopoverController } from
'@ionic/angular';
// import {  } from './'
import { PopComponent } from '../pop/pop.component';

@Component({
```

```
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  private actionSheetMsg: number;
  private fabMsg: string;

  constructor(
    private actionSheetCtrl: ActionSheetController,
    private alertCtrl: AlertController,
    private popoverCtrl: PopoverController,
  ) {}

  async presentActionSheet() {
    const actionSheet = await this.actionSheetCtrl.create({
      header: '액션시트를 통해 값을 지정해보자',
      buttons: [{
        text: 'alert 예제 확인',
        icon: 'trash',
        handler: async () => {
          const alert = await this.alertCtrl.create({
            header: 'Alert 눌렀냐?',
            subHeader: 'Alert 예제 공부하나부네~',
            message: '경고메세지야!',
            buttons: [
              {
                text: '0 나와라!',
                handler: () => {
                  this.actionSheetMsg = 0;
                }
              },
              {
                text: '취소',
                role: 'cancel',
              }
            ]
          });
```

```
        await alert.present();
      }
    }, {
      text: '1',
      icon: 'share',
      handler: () => {
        this.actionSheetMsg = 1;
      }
    }, {
      text: '2',
      icon: 'arrow-dropright-circle',
      handler: () => {
        this.actionSheetMsg = 2;
      }
    }, {
      text: '3',
      icon: 'heart',
      handler: () => {
        this.actionSheetMsg = 3;
      }
    }, {
      text: 'Cancel',
      icon: 'close',
      role: 'cancel',
      // handler: () => {
      //   console.log('Cancel clicked');
      // }
    }]
  });
  await actionSheet.present();
}

fab(btnString: string) {
  this.fabMsg = btnString;
}

async presentPopover(ev: any) {
  const popover = await this.popoverCtrl.create({
    component: PopComponent,
```

```
      event: ev,
      translucent: true
    }).then(popoverData => {
      popoverData.present();
    }).catch(err => {
      console.log(err);
    });
  }

}
```

Import your component in both the ts file you are using, and app module, and put your library in declaration and entry component

Ion-menu
In html

In ts

c. BitCoin$ Project

Congratulation! You've almost finished the fundamentals to make an app! Let us go for the first journey!
축하드립니다! 앱을 만들기 위한 기초들은 거의 끝나셨어요! 첫번째 여행을 떠나볼까요?

Let's make an app that shows bitcoin price in real time! Now, some readers might wonder how to do that because this book has not really covered necessary things such as internet-related functions. Don't worry! All the things that you need to do for this project that has not yet covered will be provided with explanation.
실시간 비트코인 가격을 보여주는 앱을 만들어 볼까요? 몇몇 독자들은 인터넷 관련된 부분을 한번도 배운 적이 없기에 겁부터 내실 것입니다. 걱정 마셔요! 지금까지 배우지 않은 부분은 프로젝트에서 다 설명해드릴 것이랍니다.

Click the link below, and you'll see a json object printed on the screen.
아래 링크를 클릭하시고 인터넷 브라우저에 프린트된 json object를 보세요.

https://apiv2.bitcoinaverage.com/indices/global/ticker/ETHUSD

As you can see, json object has a form, {}. Now, check out how they have something in quotation, a colon and a number or another {}. For instance, "ask" : __number, … , open: { __something }.

보시면 알겠지만, json object는 중괄호 포맷을 가지고 있습니다. 또 자세히 보시면, 쌍따옴표로 된 부분, 콜론으로 된 부분 그 후에 숫자 또는 또 다른 중괄호가 있습니다. 예를 들면, “ask”: __number가 있고 또 open: { __something }이 있습니다.

d. Navigation

Ion-modal & Ion-nav
In html

In ts

Ion-route & Ion-router
In html

In ts

Ion-tabs
In html

In ts

e. For better User Experience

Ion-anchor
Ion-infinite-scroll
Ion-loading & ion-progressbar & ion-skeleton-text & ion-spinner
Ion-refresher
Ion-reorder
Ion-slides
Ion-toast

f. Group & Input

Ion-card
Ion-input & form
Ion-textarea
Ion-item
Ion-list
Ion-searchbar

g. Asthetics

Ion-badge
Ion-ripple-effect
Ion-chip
Ion-grid
Ion-icon (xd)
Ion-avatar
Ion-img & ion-thumbnail

h. D
i. D
j.