Vue.js 시작하기

2020-03-27

이승진

**학습목표**

Vue.js 시작하기

firebase hosting 서비스 이용하기

firebase realtime database 이용하기

**목차**

[1. Vue.js 소개 2](#_Toc37683186)

[2. firebase 서버 프로젝트 생성 3](#_Toc37683187)

[1) firebase 서비스 3](#_Toc37683188)

[2) firebase 서버 프로젝트 생성 3](#_Toc37683189)

[3) firebase 프로젝트에 웹 앱 추가 6](#_Toc37683190)

[4) realtime database 활성화 8](#_Toc37683191)

[5) firebase CLI 11](#_Toc37683192)

[3. 소스코드 프로젝트 생성 14](#_Toc37683193)

[1) 프로젝트 디렉토리 생성 14](#_Toc37683194)

[2) 프로젝트 설정 파일(package.json) 생성 14](#_Toc37683195)

[3) eslint 설정 파일 생성 15](#_Toc37683196)

[4) firebase 설정 파일 생성 17](#_Toc37683197)

[4. hello world 19](#_Toc37683198)

[1) hello.html 생성 19](#_Toc37683199)

[2) 프로젝트 실행 20](#_Toc37683200)

[5. hello2.html 21](#_Toc37683201)

[1) hello2.html 생성 21](#_Toc37683202)

[2) 실행 23](#_Toc37683203)

[3) 서버 데이터 변경 24](#_Toc37683204)

[6. 서버에 설치 25](#_Toc37683205)

[1) firebase hosting local server 종료 25](#_Toc37683206)

[2) 업로드 25](#_Toc37683207)

[3) 실행 26](#_Toc37683208)

[4) 무료 27](#_Toc37683209)

[7. 과제 28](#_Toc37683210)

[1) hello3.html 28](#_Toc37683211)

[2) github repository 28](#_Toc37683212)

# Vue.js 소개

### Application Framework Class Library

앱 개발에 필요한 골격에 해당하는 클래스들을 모아 놓은 라이브러리.

줄여서 Application Framework 이라고 부른다.

### Javascript Frontend Application Framework

javascript 언어로 frontend app을 개발하기 위한 라이브러리.

대표적인 프레임웍은 다음과 같다.

- React.js : 완성도가 가장 높고, 진입 장벽도 높다 (어렵다).

- Angular.js

- Vue.js : 진입 장벽이 가장 낮다 (쉽다).

전문 frontend 개발자가 목표라면 => React.js

backend 개발자인데 frontend도 할 줄 알아야 한다 => Vue.js

### Backend Framework

정보시스템의 경우 Java 언어 + Spring Framework 기술이 대세.

# firebase 서버 프로젝트 생성

## firebase 서비스

모바일 애플리케이션이나 웹 애플리케이션에 필요한 서버 기능을 제공하는 google의 서비스이다.

개발과 테스트 용도로는 무료로 사용할 수 있다.

사용자가 많아지면 비용을 지불해야 하는데, 유사한 다른 서비스들에 비해 저렴하다고 알려져 있다.

### firebase hosting 서비스

javascript로 개발한 프로젝트를 서버에 설치해서 운영해 주는 서비스이다.

### firebase realtime database

애플리케이션 데이터를 서버에 저장하기 위한 서비스이다.

관계형 데이터베이스는 아니다.

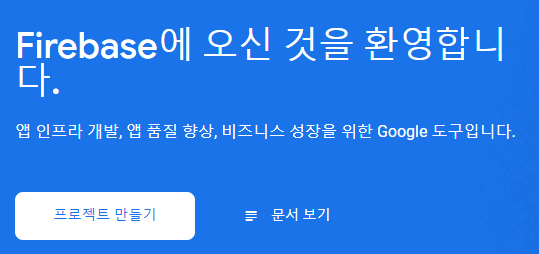
key, value 구조로 데이터를 저장하고 조회하는 방식이다.

## firebase 서버 프로젝트 생성

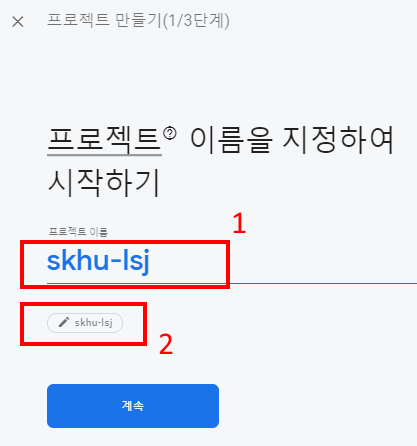
firebase 서버의 서비스는, firebase 프로젝트 단위로 제공된다.

따라서 먼저 firebase 서버에 프로젝트를 생성해야 한다.

<https://console.firebase.google.com/>



프로젝트 만들기 클릭



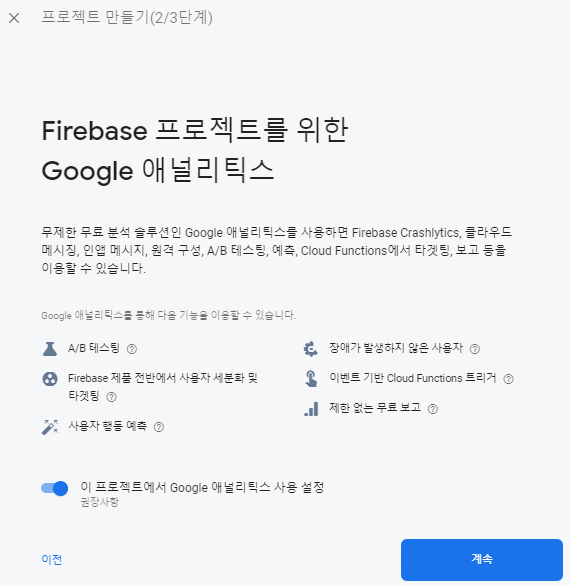
(1) 프로젝트 이름에 자신만의 프로젝트 이름을 입력하자.

기억하기 쉬운 이름이 바람직하다.

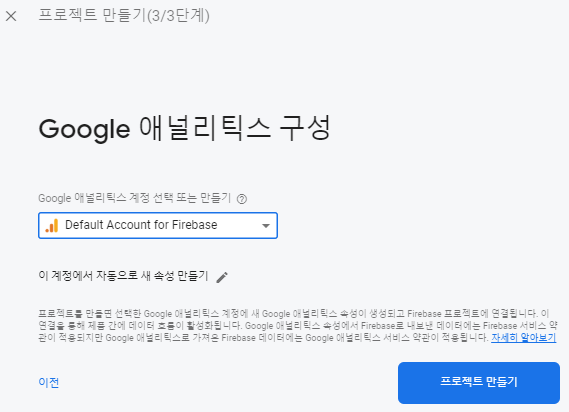
(2) 프로젝트 ID도 프로젝트 이름과 동일하게 입력하자.

프로젝트 ID는 유일무이해야 한다.

계속 클릭



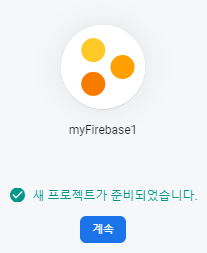
계속 클릭



Default Account for Firebase

항목을 선택하고

프로젝트 만들기 버튼 클릭



계속 클릭

google firebase 서버에 자신만의 프로젝트가 생성되었다.

## firebase 프로젝트에 웹 앱 추가

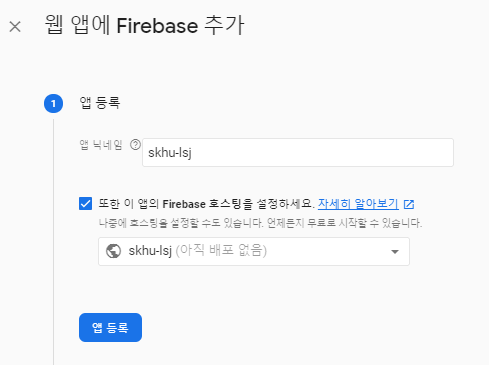
내가 만든 웹 앱이 firebase 서비스를 사용할 수 있으려면,

firebase 프로젝트에 앱을 등록해야 한다.



위 화면의 버튼들은 왼쪽부터 순서대로, iOS 앱 등록, 안드로이드 앱 등록, 웹 앱 등록 버튼이다.

웹 앱 등록 버튼 클릭



앱 닉테임에 적당한 이름을 입력하자. 프로젝트 이름과 동일하게 입력하는 것도 좋다.

'이 앱의 firebase 호스팅을 설정하세요' 항목을 체크하고

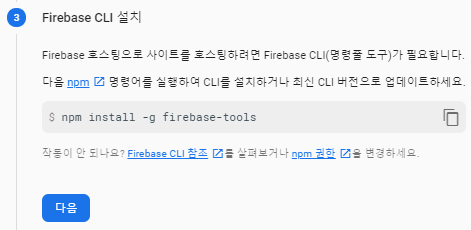
앱 등록 클릭



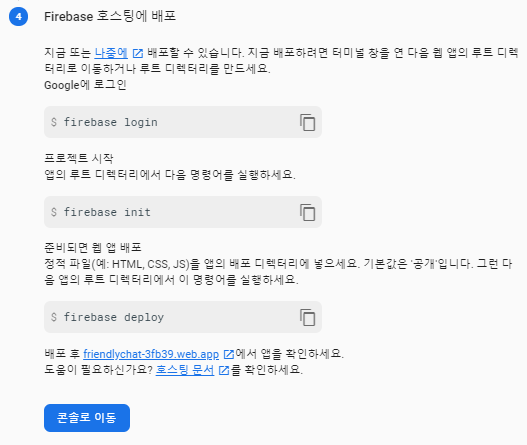
다음 버튼 클릭

|  |
| --- |
| <!-- The core Firebase JS SDK is always required and must be listed first -->  <script src="/\_\_/firebase/7.13.1/firebase-app.js"></script>  <!-- TODO: Add SDKs for Firebase products that you want to use       https://firebase.google.com/docs/web/setup#available-libraries -->  <script src="/\_\_/firebase/7.13.1/firebase-analytics.js"></script>  <!-- Initialize Firebase -->  <script src="/\_\_/firebase/init.js"></script> |

javascript로 구현한 웹 앱에서, firebase 서비스에 연결하기 위해 필요한 태그들이다.



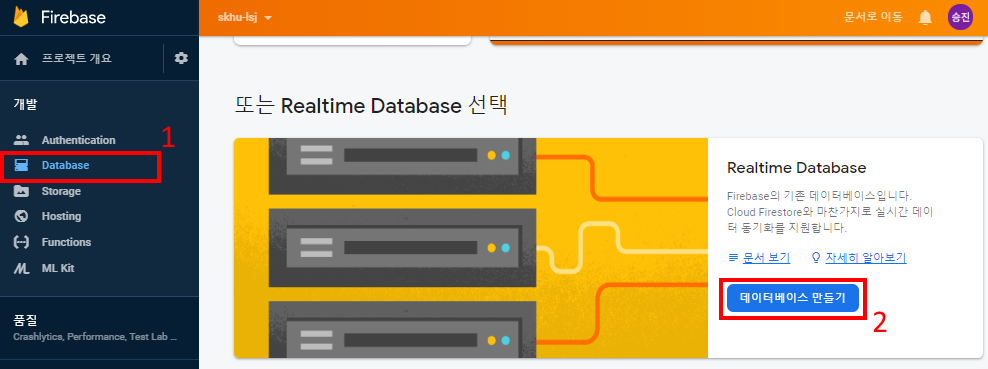
다음 버튼 클릭



콘솔로 이동 버튼 클릭

## realtime database 활성화

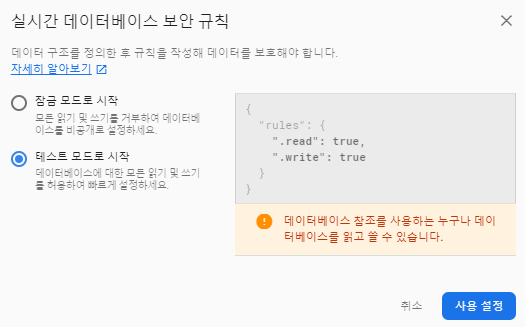
firebase 서버가 제공하는 realtime database 서비스를 활성화하자.



(1) Database 메뉴를 클릭하고

나타는 웹페이지를 아래로 내려서

(2) Realtime Database 항목 아래의 '데이터베이스 만들기' 버튼을 클릭하자.

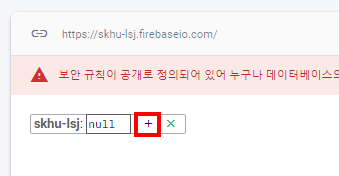


빠른 테스트를 위하여,

테스트 모드로 시작 항목을 선택하고

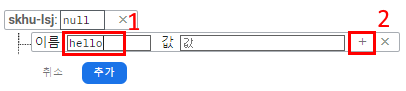
사용 설정 버튼 클릭

### 데이터 항목 입력하기



+ 버튼(add child 버튼)을 클릭하자.

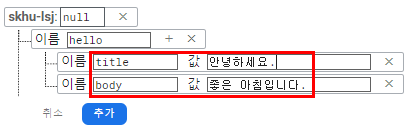
skhu-lsj 항목 아래에 자식 항목이 한 개 생성된다.



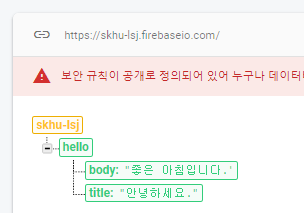
(1) hello를 입력하고

(2) + 버튼(add child 버튼)을 두 번 클릭.

hello 항목 아래에 자식 항목이 두 개 생성된다.



위와 같이 입력하고 추가 버튼 클릭



skhu-lsj 프로젝트의 firebase realtime database에

hello 항목을 생성하였다.

이 항목의 값은 객체이다.

이 객체에는 body 속성과, title 속성이 있다.

hello.title 속성값은 "안녕하세요"

hello.body 속성값은 "좋은 아침입니다."

## firebase CLI

### firebase hosting 서비스

내가 개발한 웹 앱을 firebase 서버에 설치하고 운영할 수 있다.

### firebase CLI

firebase hosting 서비스를 사용하기 위한 명령들이다.

firebase CLI 명령을 사용하여 다음 작업들을 할 수 있다.

- 웹 앱 프로젝트를 firebase hosting 서비스에 설치하여 운영.

- 웹 앱 프로젝트를 firebase hosting 서비스와 동일한 환경으로 내 PC 에서 실행하여 테스트.

### firebase CLI 설치

명령 프롬프트에서 아래 명령을 실행하자.

|  |
| --- |
| npm install -g firebase-tools |

### 설치 검증

|  |
| --- |
| firebase --version |

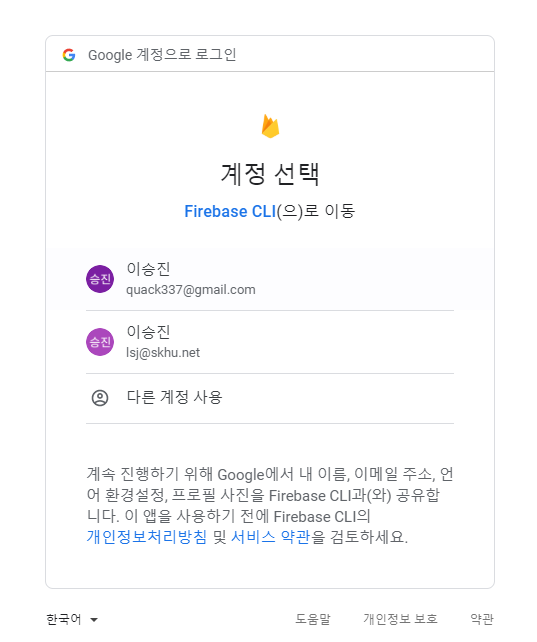
|  |
| --- |
|  |

### firebase login

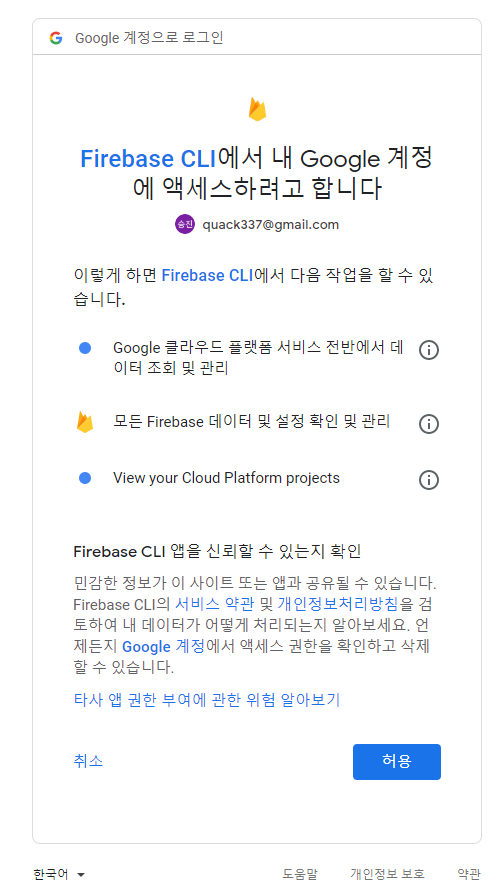
firebase CLI에서 google 계정에 로그인해야 한다.

|  |
| --- |
| firebase --interactive login |

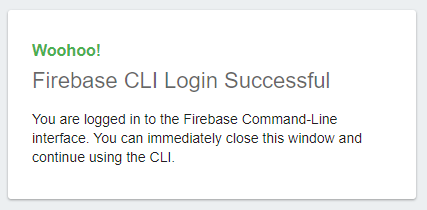
중간에 프롬프트에서 Y를 입력한다.



내 google 계정 선택



허용 클릭



|  |
| --- |
|  |

로그인 성공

### firebase 프로젝트 목록 보기

|  |
| --- |
| firebase projects:list |

|  |
| --- |
|  |

firebase 서버에 생성한 프로젝트 목록을 보여준다.

# 소스코드 프로젝트 생성

vue.js 예제들을 실행해 보기위한 간단한 웹앱 프로젝트를 생성하자.

이 웹앱 프로젝트를 firebase 서버의 hosting 서비스에 설치할 예정이다.

## 프로젝트 디렉토리 생성

프로젝트를 만들기에 적당한 디렉토리로 이동하자.

|  |
| --- |
| cd /c/PJ/vue |

적당한 이름의 프로젝트 디렉토리를 만들자.

|  |
| --- |
| mkdir skhu-lsj |

자신만의 프로젝트 이름을 사용하자.

firebase 프로젝트 이름과 동일한 이름을 사용해도 좋다.

|  |
| --- |
|  |

## 프로젝트 설정 파일(package.json) 생성

javascript 프로젝트(node.js 프로젝트)의 설정 파일은 package.json 파일이다.

아래 명령을 실행하여 이 파일을 생성하자.

|  |
| --- |
| npm init -yes |

주의 사항: 프로젝트 폴더에서 이 명령을 실행해야 한다.

|  |
| --- |
|  |

## eslint 설정 파일 생성

eslint는 javascript 소스 코드의 오류를 찾아준다.

프로젝트에서 eslint의 기능을 이용할 수 있으려면, eslint 설정 파일이 필요하다.

프로젝트 폴더에 eslint 설정 파일을 만들자.

eslint 설정 파일은 .eslintrc 이다.

프로젝트 폴더에서 아래 명령을 실행하자.

|  |
| --- |
| eslint --init |

visual studio code의 터미널 창의 bash 쉘에서 위 명령을 실행해야 한다.

그렇지 않으면, 화살표 키가 제대로 작동하지 않을 수 있다.

|  |
| --- |
|  |

위 아래 화살표 키를 눌러서 'To check syntax and find problems' 항목을 선택하고

엔터키 입력

check syntax : 문법 오류를 검사한다

find problems : 흔한 실수를 검사한다

enforce code style: 코드 스타일을 검사한다

|  |
| --- |
|  |

Javascript modules (import/export) 항목을 선택하고 엔터키 입력

|  |
| --- |
|  |

Vue.js 항목을 선택하고 엔터키 입력. (Vue.js 앱을 만들 것이므로)

|  |
| --- |
|  |

N을 입력하고 엔터키 입력

|  |
| --- |
|  |

Browser 항목에 별표가 표시된 상태에서 (Vue.js 앱은 웹브라우저에서 실행되므로)

엔터키 입력

|  |
| --- |
|  |

JSON 항목을 선택하고 엔터

(생성될 eslint 설정 파일의 포맷을 지정한다)

|  |
| --- |
|  |

Y를 입력하고 엔터키 입력

위 명령에 의해서, eslint의 설정 파일인 .eslintrc.json 파일 생성된다.

그리고 필요한 javascript 모듈들이, 프로젝트 폴더의 node\_modules 폴더 아래에 다운로드 된다.

## firebase 설정 파일 생성

우리의 웹 앱 프로젝트에서 firebase 서비스들을 이용할 수 있으려면, 먼저 설정 파일들을 만들어야 한다.

firebase cli 명령을 사용하여 설정 파일들을 만들자.

프로젝트 폴더에서 아래 명령을 실행하자.

|  |
| --- |
| firebase init |

|  |
| --- |
|  |

Y 입력 후 엔터 키 입력

|  |
| --- |
|  |

이용하려는 firebase 서비스 종류를 선택하는 화면이다.

화살표 키와 스페이스 키를 눌러서 위와 같이 선택하고

엔터 키 입력

|  |
| --- |
|  |

위와 같이 선택하고

엔터 키 입력

|  |
| --- |
|  |

firebase 서버 프로젝트 이름을 선택하고 엔터 키 입력

|  |
| --- |
|  |

엔터 키 입력

|  |
| --- |
|  |

엔터 키 입력

**위 화면의 의미:**

public 폴더 아래의 파일들이 firebase 서버에 업로드 되어 설치된다

|  |
| --- |
|  |

지금 우리가 만들려고 하는 것은 single-page app 이 아니므로 N 입력하고 엔터

|  |
| --- |
|  |

완료

다음과 같은 파일들이 자동으로 생성되었다.

|  |  |
| --- | --- |
| 파일 | 설명 |
| public/404.html | 잘못된 URL 요청되었을 때 출력될 페이지 |
| public/index.html | 시작 페이지 |
| firebase.json | firebase 서비스 설정 파일 |
| .firebaserc | firebase 서버 프로젝트 정보 |
| .gitignore | git 설정 파일 |

# hello world

## hello.html 생성

### public/hello.html

|  |
| --- |
| <html>  <head>    <meta charset="utf-8" />    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>    <div id="app">      <h1>{{ title }}</h1>      <p>{{ body }}</p>    </div>    <script type="text/javascript">      var app = new Vue({        el: '#app',        data: {          title: 'hello world',          body: 'good morning'        }      })     </script>  </body>  </html> |

<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>

vue.js 엔진에 해당하는 javascript 파일에 대한 링크이다.

이 파일의 소스코드가 이 웹브라우저 창에서 실행된다.

즉 vue.js 엔진이 이 창에서 실행된다.

var app = new Vue({ 생략 })

Vue 객체를 생성하여 전역 변수 app에 대입한다.

이 객체가 Vue 앱에 해당하는 객체이다.

Vue 생성자 파라미터는 javascript 객체이다. (노란색으로 칠한 부분)

이 객체에, Vue 앱의 기능을 구현해야 한다.

el: '#app'

Vue 앱의 기능이 실행될 태그의 id를 지정한다.

  Vue 앱의 기능은 <div id="app"> 태그 내부에서만 실행된다.

data: {

title: 'hello world',

  body: 'good morning'

}

앱의 속성을 선언한 코드이다. (멤버 변수)

<h1>{{ title }}</h1>

title 속성값이 여기에 출력된다.

<p>{{ body }}</p>

body 속성값이 여기에 출력된다.

## 프로젝트 실행

프로젝트를 firebase 서버의 hosting 서비스에 설치하기 전에,

내 PC 에서 먼저 실행하고 테스트할 수 있다.

프로젝트를 firebase hosting local server에서 실행하면,

firebase hosting 서비스와 동일한 환경으로 내 PC에서 프로젝트가 실행된다.

프로젝트 디렉토리에서 다음 명령을 실행하자.

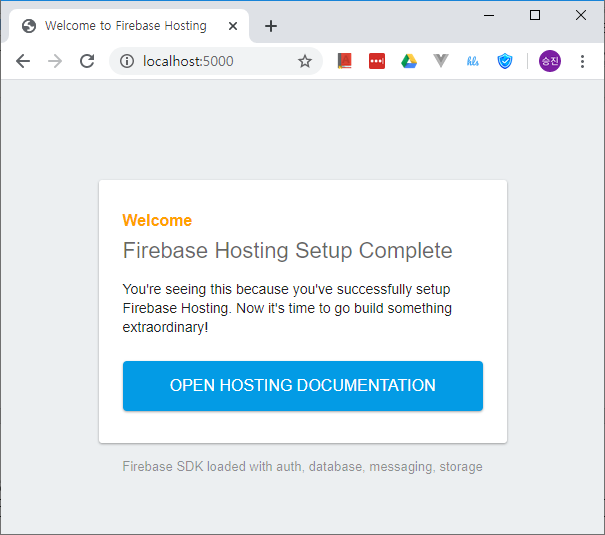
|  |
| --- |
| firebase serve --only hosting |

|  |
| --- |
|  |

firebase hosting local server 가 시작되었다.

위 화면의 URL에 주목하자.

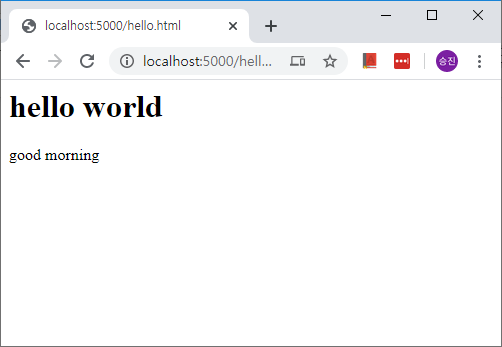
웹브라우저에서 이 URL을 열자.



시작 페이지(자동으로 생성된 public/index.html 파일)의 내용이 출력된다.

public/hello.html 파일을 실행하자.

|  |
| --- |
| <http://localhost:5000/hello.html> |



title 속성 값과, body 속성 값이 출력되었다.

# hello2.html

## hello2.html 생성

### public/hello2.html

|  |
| --- |
| <html>  <head>    <meta charset="utf-8" />    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>    <script defer src="/\_\_/firebase/7.13.1/firebase-app.js"></script>    <script defer src="/\_\_/firebase/7.13.1/firebase-database.js"></script>    <script defer src="/\_\_/firebase/init.js"></script>  </head>  <body>    <div id="app">      <h1>{{ title }}</h1>      <p>{{ body }}</p>    </div>    <script type="text/javascript">      var app = new Vue({        el: '#app',        data: {          title: 'hello world',          body: 'good morning'        }      })      window.onload = function() {        let helloRef = firebase.database().ref("hello");        helloRef.on('value', (snapshot) => {          let hello = snapshot.val();          app.title = hello.title;          app.body = hello.body;        });      };     </script>  </body>  </html> |

<script defer src="/\_\_/firebase/7.13.1/firebase-app.js"></script>

<script defer src="/\_\_/firebase/7.13.1/firebase-database.js"></script>

<script defer src="/\_\_/firebase/init.js"></script>

firebase 서비스를 실행하기 위해 필요한 javascript 파일들에 대한 링크이다.

위 javascript 파일들이 이 웹페이지에서 실행된다.

### defer 키워드

위 script 태그에서 defer 키워드에 주목하자.

defer 키워드가 없으면, script 태그가 참조하는javacsript 파일이 즉시 다운로드되고 실행된다.

defer 키워드가 있으면, 다운로드와 실행이 뒤로 늦춰진다.

이 페이지의 내용이 웹브라우저에 다 출력된 후, 다운로드 되고 실행된다.

시간이 걸리는 작업을 뒤로 미뤄서, 웹페이지의 내용이 좀 더 빨리 화면에 나타나는 효과가 있다.

window.onload = function() {

... 생략 ...

};

window 객체의 onload 속성에 함수를 대입한다.

windows 객체는 현재 웹브라우저 창에 해당하는 객체이다.

onload 속성은 콜백 함수를 등록하기 위한 속성이다.

현재 웹브라우저 창의 내용의 로드가 완료되면, onload 속성에 등록된 함수가 자동으로 호출된다.

웹브라우저 창의 내용이 전부 로드된 후 실행될 코드를 구현할 때,

위와 같이 구현한다.

let helloRef = firebase.database().ref("hello");

firebase 데이터베이스의 "hello" 항목에 대한 원격 참조 객체를 구해서, helloRef 지역 변수에 대입한다.

helloRef.on('value', (snapshot) => {

... 생략 ...

});

helloRef 원격 참조 객체가 참조하는 데이터베이스 항목의 값이 변경되자마자

하늘색으로 칠한 부분의 화살표 함수가 실행된다. **(snapshot) => { ... 생략 ... }**

화살표 함수의 파라미터 값은 데이터베이스 항목의 값이다.

이 값이 firebase 서버로부터 웹브라우저 창까지 전달된다.

데이터베이스 항목의 값이 변경되지 않더라도,

현재 웹페이지가 웹브라우저 창에 로드될 때, 최초 한 번 이 화살표 함수가 호출된다.

요약 하자면,

helloRef 원격 참조 객체가 참조하는 데이터베이스 항목의 값이

이 화살표 함수의 파라미터 값으로 전달된다.

이 페이지가 시작되자 마자 한 번 전달되고,

DB에서 그 항목의 값이 바뀌자마자 또 전달된다.

(snapshot) => {

let hello = snapshot.val();

  app.title = hello.title;

  app.body = hello.body;

}

let hello = snapshot.val();

전달된 DB 항목의 값을 꺼내서 hello 지역 변수에 대입한다.

app.title = hello.title;

전달된 DB 항목의 title 속성의 값을, 앱의 title 속성에 대입한다.

title 속성 값이 바뀌면, 그 값이 {{ title }} 부분에 즉시 다시 출력된다.

app.body = hello.body;

전달된 DB 항목의 body 속성의 값을, 앱의body 속성에 대입한다.

body 속성 값이 바뀌면, 그 값이 {{ body }} 부분에 즉시 다시 출력된다.

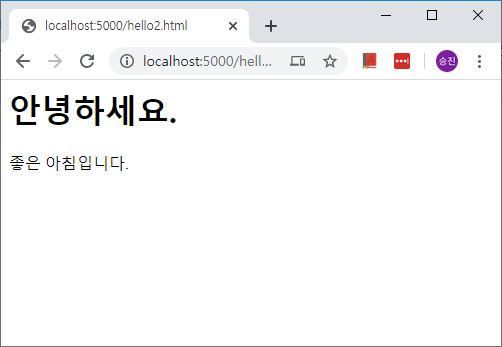
## 실행

firebase hosting local server를 재시작할 필요 없다.

수정되거나 추가된 파일이 즉시 반영된다.

웹브라우저 창에서 다음 URL을 실행하자.

|  |
| --- |
| <http://localhost:5000/hello2.html> |



data: {

title: 'hello world',

  body: 'good morning'

}

최초 title 속성과 body 속성의 값은 위와 같다.

따라서 위 페이지가 실행되었을 때, 위 값이 화면에 출력된다.

let hello = snapshot.val();

  app.title = hello.title;

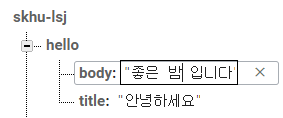
  app.body = hello.body;

그리고, DB로부터 hello 항목의 값이 도착하자 마자 위 코드가 실행되어,

title 속성과 body 속성 값이 바뀌고,

화면의 출력도 즉시 바뀐다.

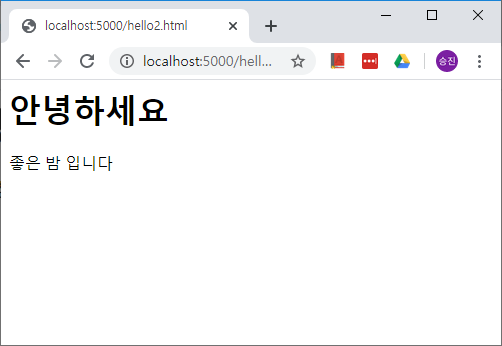
## 서버 데이터 변경



firebase 콘솔(<https://console.firebase.google.com/>)에서

위 화면과 같이 데이터 값을 수정하자.

hello2.html 웹브라우저 창의 내용이 자동으로 즉시 변경되는 것을 확인하자.



# 서버에 설치

아직 별로 구현한 기능은 없지만,

프로젝트를 firebase hosting 서비스에 업로드해서 설치하자.

## firebase hosting local server 종료

firebase hosting local server 가 실행중인 터미널에서 Ctrl+C를 눌러서

local server를 종료하자.

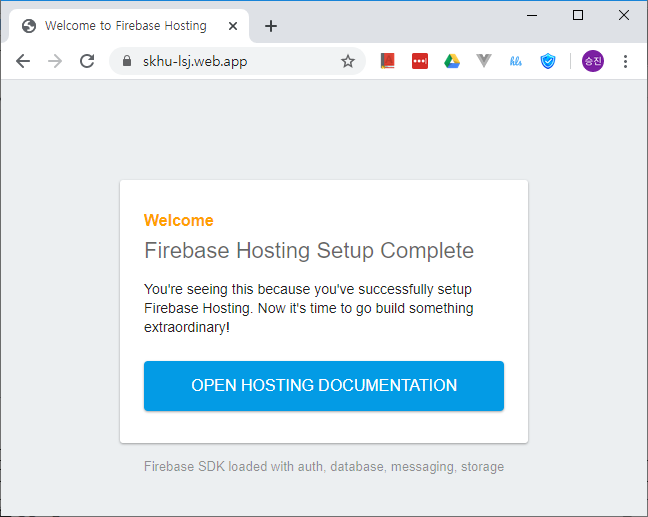
## 업로드

프로젝트 디렉토리에서 아래 명령을 실행하자.

|  |
| --- |
| firebase deploy |

|  |
| --- |
|  |

## 실행



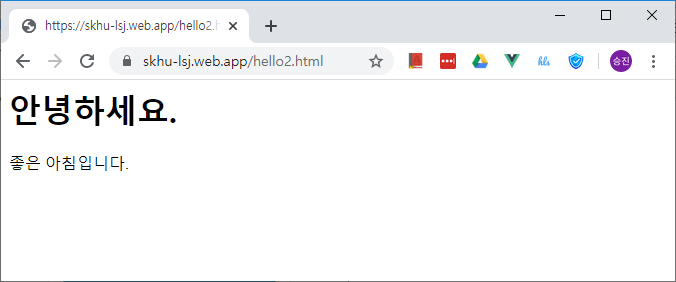
firebase hosting 서비스 URL은

firebase프로젝트ID.web.app 이다.

위 화면은 자동 생성된 index.html 파일 화면이다.

### hello2.html

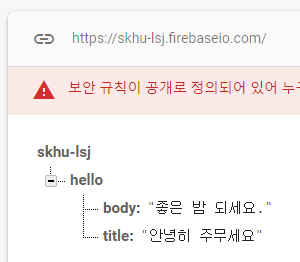
|  |
| --- |
| firebase프로젝트ID.web.app/hello2.html |



### 서버 데이터 수정

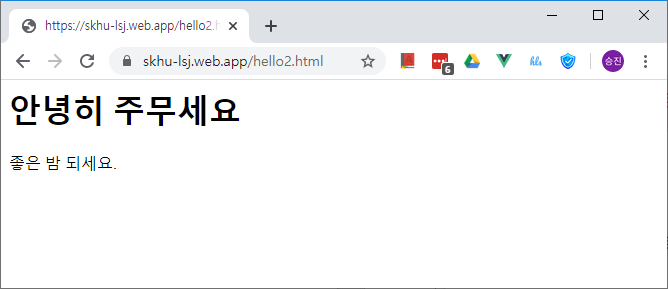
위 웹브라우처 창이 열려있는 상태에서,

서버의 hello 데이터 항목의 값을 수정하자.



위와 같이 서버의 데이터가 변경되자 마자

웹브라우저 창의 내용도 즉시 변경된다.



서버의 데이터가 변경되자 마자, 웹브라우저 창의 내용도 즉시 변경되는 이유를 생각해 보자.

그 실행 과정을 소스 코드 수준에서 분석하자.

## 무료

사용량이 많지만 않다면, firebase hosting 서비스도 무료로 사용할 수 있다.

사용자가 많아지면 돈을 내야 하지만, google 광고를 걸어서 돈을 벌 수도 있다.

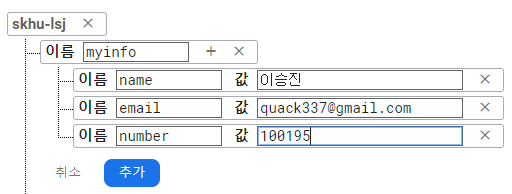
google 광고를 거는 방법도 간단하다.

광고로 버는 돈과, 사용 요금 중에서 무엇이 더 클까?

# 과제

## hello3.html

firebase realtime database에 다음과 같이 데이터 항목을 추가하자.



(1) skhu-lsj 항목 옆의 + 버튼을 클릭하면 자식 노드가 생긴다

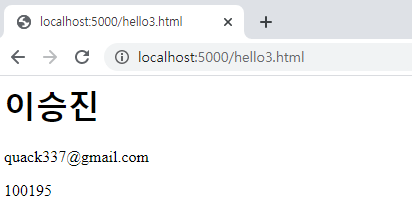
(2) 자식 노드에 myinfo 입력

(3) myinfo 노드 옆의 + 버튼을 세번 클릭하면 자식 노드가 3개 생긴다.

(4) 위 화면과 같이 입력하고

(5) 추가 버튼 클릭하여 저장

위와 같이 이름, 이메일, 학번을 입력하자.



위와 같이 실행되도록 hello3.html 파일을 구현하자.

## github repository

vue.js 과제를 제출하기 위한 프로젝트와 github repository를 새로 만들자.

이 프로젝트에 과제 파일 뿐만 아니라, 강의 예제가 들어있어도 된다.

아래 링크를 열어서 github repository를 등록하자.

월요일 수업

<https://forms.gle/wMAJhux8SgRLmYSV8>

수요일 수업

<https://forms.gle/MkAcnq4L9rLMGEnR7>

일요일 자정까지 과제와 url 둘 다 제출하자.