

시스템 분석설계 개인 포트폴리오

컴퓨터정보공학과 2-B
20192491 김동휘

목차

1. 소프트웨어 공학

2. 깃과 깃허브

3. 파이어베이스



1. 소프트웨어 공학

1. 소프트웨어의 정의
2. 소프트웨어의 특징
3. 소프트웨어 공학 개요
4. 소프트웨어 개발 프로세스
5. 소프트웨어 프로세스 모델

1. 소프트웨어 공학 - 소프트웨어의 정의

소프트웨어란?

넓은 의미로 컴퓨터 하드웨어를 가장 효율적으로 사용하기 위한 모든 기술로써 프로그램(원시코드)과 이를 이용하기 위한 여러가지 절차, 이에 관계되는 모든 문서들, 그리고 전산화된 데이터와 정보 등을 모두 포함하는 매우 포괄적인 개념이다.

1. 소프트웨어 공학 - 소프트웨어의 특징

◎ 제조가 아닌 개발

제조는 정해진 틀에 맞춰 일정하게 생산하는 것으로, 많은 인력이 필요하고 능력별 결과물 차이가 근소하다. 개발은 개인 능력 별 결과물 차이가 매우 크다.

◎ 소모가 아닌 품질 저하

하드웨어는 오래 사용하면 부품이 닳고, 고장 발생 빈도 높고 기능도 떨어지는 반면 소프트웨어는 오래 사용해도 닳지 않고, 고장 발생 빈도 낮고, 그 자체의 기능은 변함이 없다. 다만 오래 사용하는 기간 동안 발전하는 기술로 인해 다른 제품에 비해 기능이 뒤쳐질 수 있다.

1. 소프트웨어 공학 - 소프트웨어 공학 개요

- 정의 -

품질 좋은 소프트웨어를 경제적으로 개발하기 위해 계획을 세우고, 개발하며, 유지 및 관리하는 전 과정에서 공학, 과학 및 수학적 원리와 방법을 적용하여 필요한 이론과 기술 및 도구들에 관해 연구하는 학문

- 목표 -

개발 과정에서의 생산성 향상 및 사용자 만족을 위한 고품질의 소프트웨어 생산

1. 소프트웨어 공학 - 소프트웨어 개발 프로세스

◎ 소프트웨어 프로세스 모델의 정의

- 소프트웨어 개발 생명 주기 (SLDC, Software Development Life Cycle)
- 소프트웨어를 어떻게 개발할 것인가에 대한 전체적인 흐름을 체계화한 개념
- 개발 계획 수립부터 최종 폐기 때까지의 전 과정을 다룸
- 순차적인 단계로 이루어짐

◎ 소프트웨어 프로세스 모델의 목적

- 공장에서 제품을 생산하듯이 소프트웨어 개발의 전 과정을 하나의 프로세스로 정의
- 주어진 예산과 자원으로 개발하고 관리하는 방법을 구체적으로 정의
- 고품질의 소프트웨어 제품 생산을 목적으로 함

1. 소프트웨어 공학 - 소프트웨어 개발 프로세스

소프트웨어 개발 생명 주기 (SLDC, Software Development Life Cycle)

: 계획 단계에서 유지보수 단계에 이르기까지 일어나는 일련의 과정

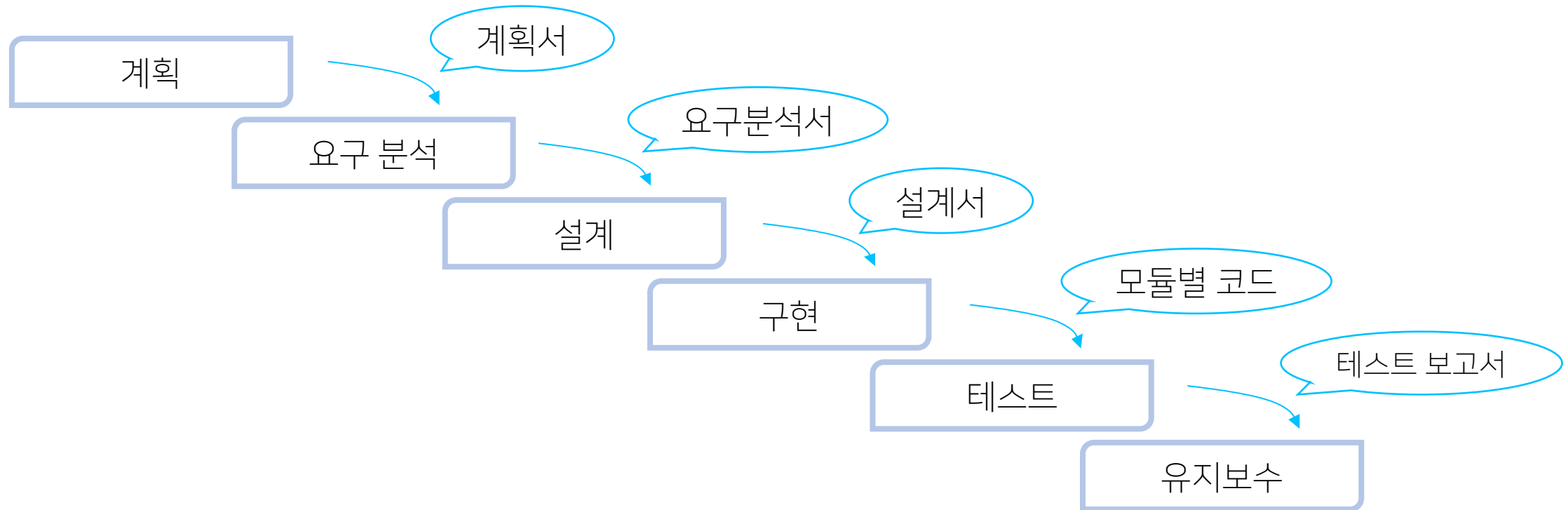


1. 소프트웨어 공학 - 소프트웨어 프로세스 모델

- 폭포수 모델

폭포수 모델 (waterfall model)

: 소프트웨어의 개발 과정을 요구분석, 설계, 구현, 테스트, 유지보수의 단계로 구분하여 순차적으로 진행하는 프로세스 모델이다.



1. 소프트웨어 공학 - 소프트웨어 프로세스 모델

- 폭포수 모델

폭포수 모델의 장점

- 관리가 용이하다.
- 체계적인 문서화 작업이 이뤄진다.
- 요구사항의 변화가 적은 프로젝트에 적합하다.

폭포수 모델의 단점

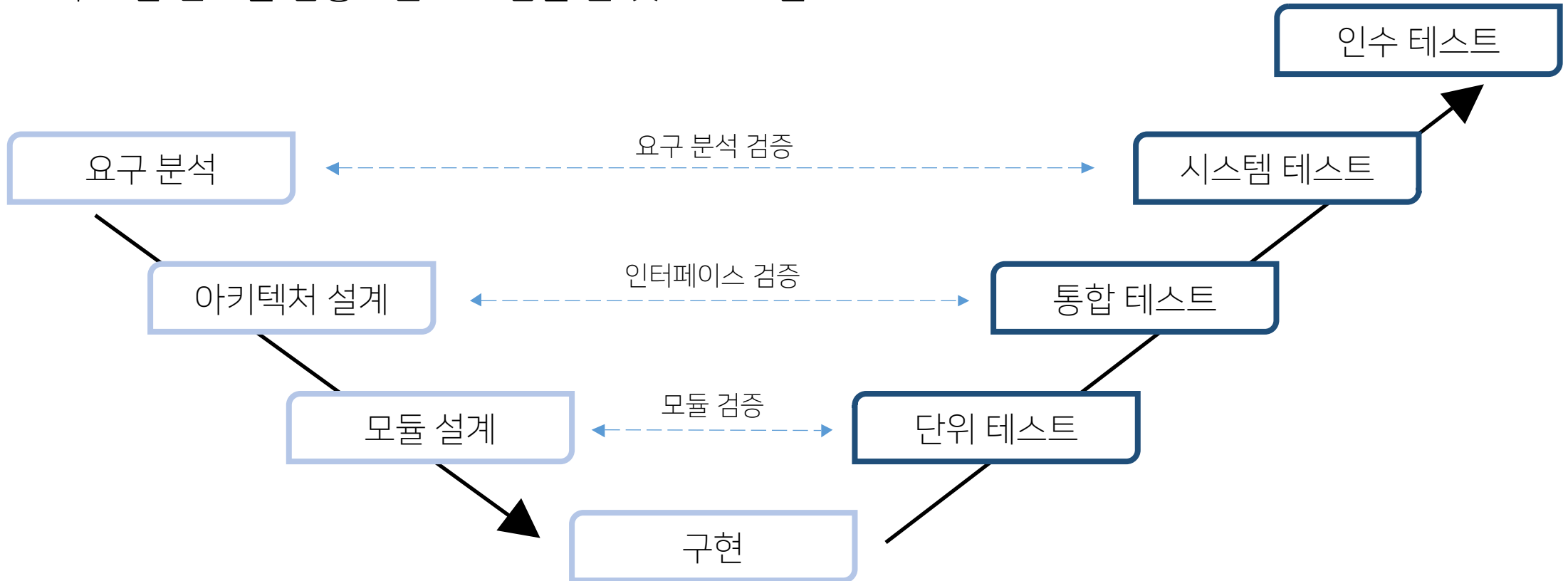
- 각 단계는 앞 단계가 완료되어야 수행할 수 있다.
- 이미 구현 단계가 진행되고 있다면 요구에 대한 변경을 수용하기 어렵다.
- 각 단계의 결과물이 완벽한 수준으로 작성되어야 다음 단계에 오류를 넘겨주지 않는다.
- 프로토타입이 없기 때문에 사용자가 중간에 가시적인 결과를 볼 수 없어 답답해할 수 있다.

1. 소프트웨어 공학 - 소프트웨어 프로세스 모델

- V 모델

V 모델 (V-model)

: 폭포수 모델 + 테스트(검증) 단계 추가 확장한 모델이다. 산출물 중심의 폭포수 모델과는 다르게 각 개발 단계를 검증하는데 초점을 둔 것이 V 모델이다.

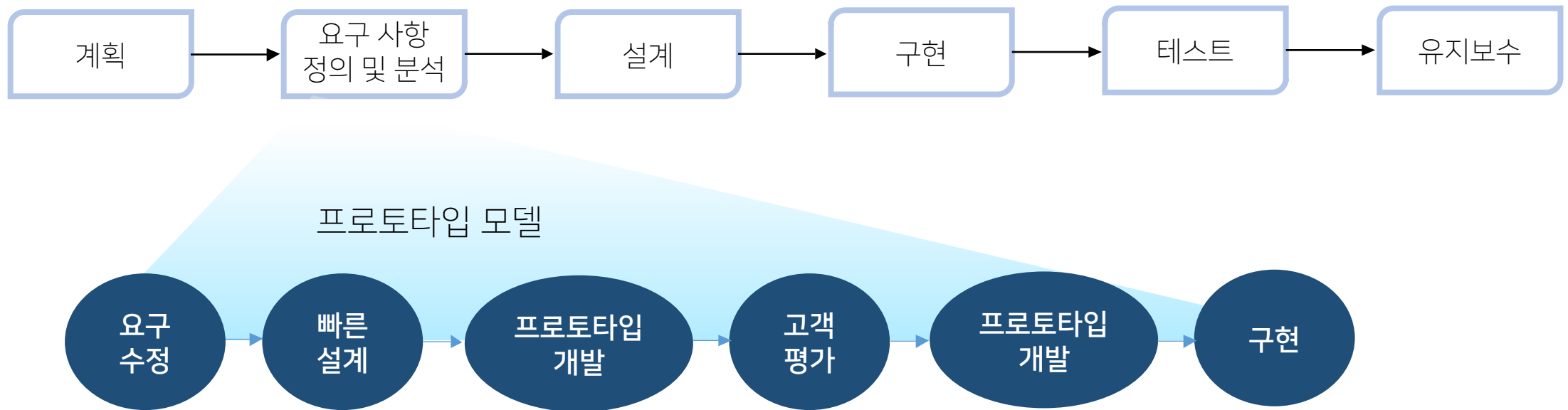


1. 소프트웨어 공학 - 소프트웨어 프로세스 모델

- 프로토타입 모델

프로토타입 모델

: 프로토타입은 대량 생산에 앞서 미리 제작해 보는 원형 또는 시제품으로, 제작물의 모형이란 뜻이다. 정식 절차에 따라 완전한 소프트웨어를 만들기 전에 사용자의 요구를 받아 일단 모형을 만들고 이 모형을 사용자와 의사소통하는 도구로 활용한다. (EX. 아파트 모델하우스)



1. 소프트웨어 공학 - 소프트웨어 프로세스 모델

- 프로토타입 모델

프로토타입 모델의 개발절차

1) 요구사항 정의 및 분석

- 1차 개략적인 요구 사항 정의 후 2차, 3차, . . . n차를 반복하면서 최종 프로토타입 개발

2) 프로토타입 설계

- 완전한 설계 대신, 사용자와 대화할 수 있는 수준의 설계
- 입출력 화면을 통한 사용자 인터페이스 중심 설계

3) 프로토타입 개발

- 완전히 동작하는 완제품을 개발하는 것이 아니며, 입력 화면을 통한 사용자의 요구 항목 확인
- 출력 결과를 통해 사용자가 원하는 것인지 확인

4) 사용자에게 의한 프로토타입 평가

- 프로토타입 평가 -> 추가 및 수정 요구 -> 프로토타입 개발 -> 프로토타입 평가



2. 깃과 깃허브



1. 깃과 깃허브 개요
2. 깃의 주요 개념
3. 깃허브 용어 정리
4. 깃허브 프로젝트 기여 과정

2. 깃과 깃허브 - 깃과 깃허브 개요



깃(Git)이란?

- 컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 분산 버전 관리 시스템이다.
- 소프트웨어 개발에서 소스 코드 관리에 주로 사용되지만 어떠한 집합의 파일의 변경사항을 지속적으로 추적하기 위해 사용될 수 있다.



깃허브(GitHub)란?

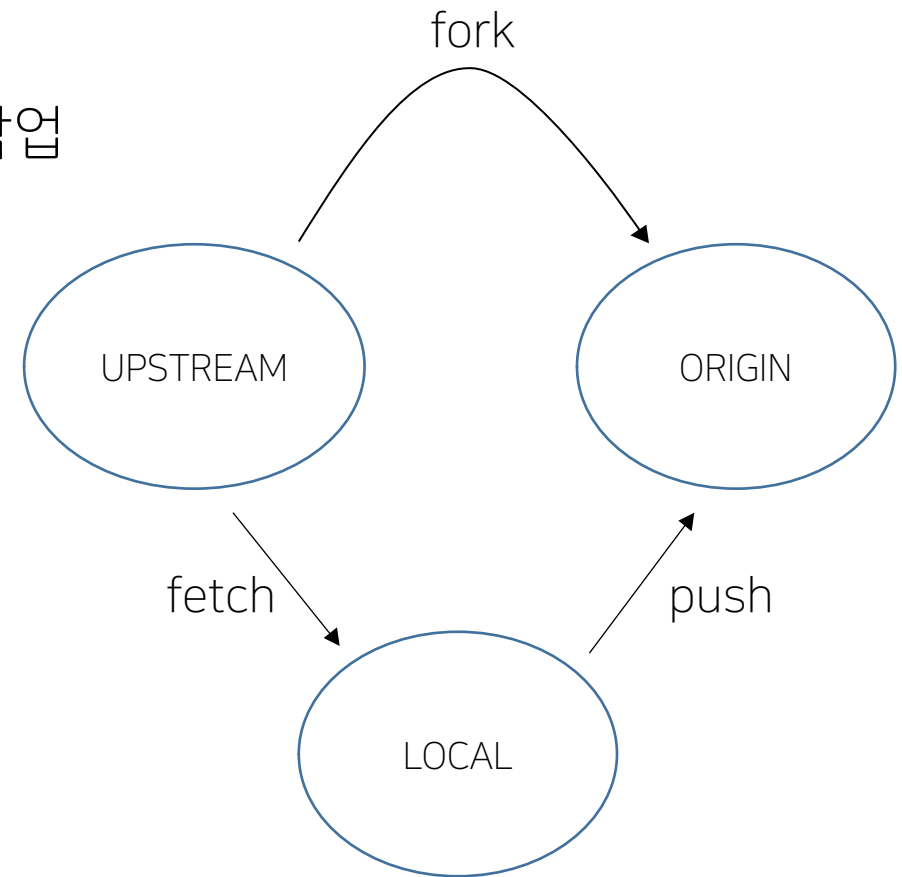
- 버전을 관리하기 위한 서버 저장소 및 프로젝트 개발을 위한 협업 관리 서비스이다.
- 깃(Git)이 텍스트 명령어 입력 방식인데 반해, 깃허브는 그래픽 유저 인터페이스를 제공한다.

2. 깃과 깃허브 - 깃(Git)의 주요 개념

- **merge** : 한 branch에서 완성한 작업을 다른 branch에 병합하기
- **tag** : 특정 이력을 가지는 commit에 대한 참조
- **pull request** : 완료한 작업을 다른 사람이 리뷰하고 병합하도록 요청하기
- **issue** : 기능에 대한 논의, 버그 추적하기
- **wiki** : 링크들을 연결해 웹페이지 만들기
- **push** : 내 컴퓨터 로컬에 저장되어 있던 버전정보를 서버(Git 저장소)에 올리기
- **pull** : Git 저장소 서버로부터 내 컴퓨터 로컬로 버전 정보 전체를 가져오기

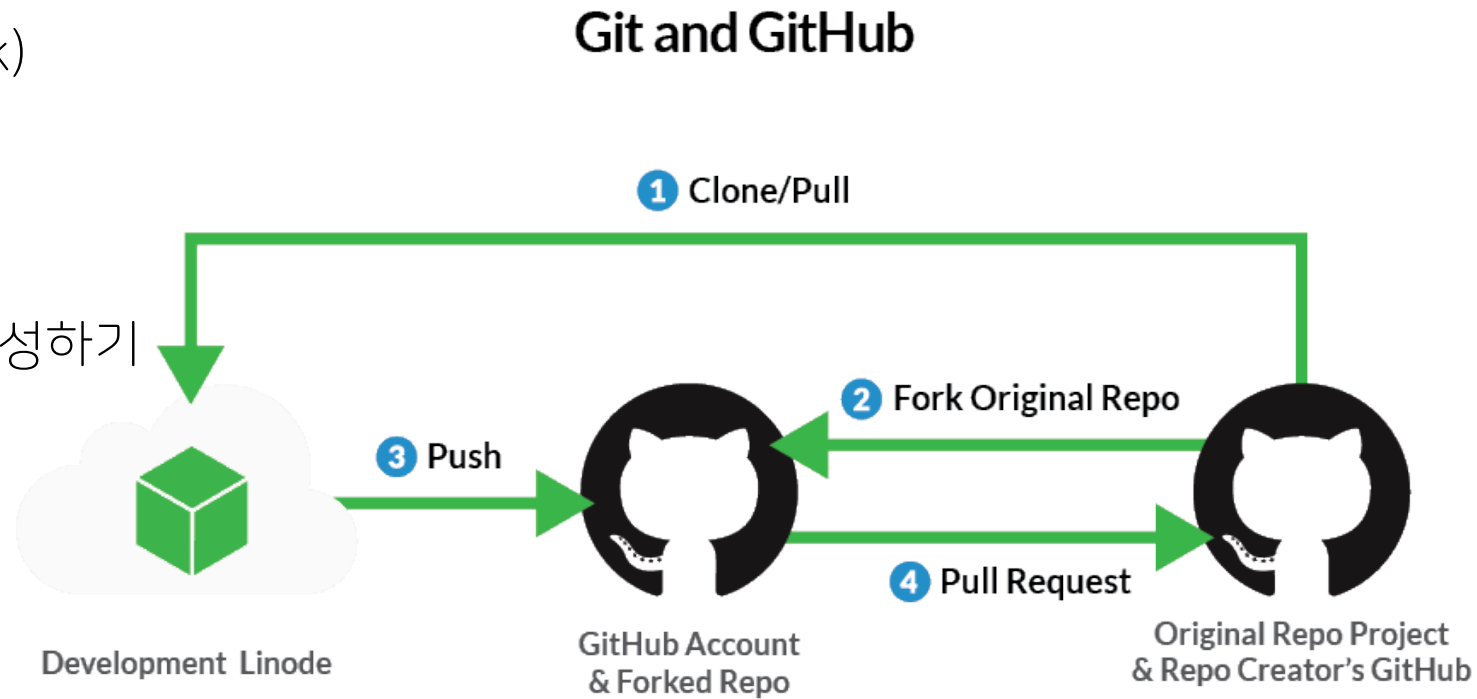
2. 깃과 깃허브 - 깃허브(GitHub) 용어 정리

- **fork** : 다른 깃허브 저장소(오픈 소스 프로젝트)를 복사하는 작업
- **upstream** : 깃허브 안에 오픈되어 있는 모든 저장소
- **origin** : 깃허브에서 내 계정에 있는 fork한 저장소
- **local** : 내가 자신의 컴퓨터에 복사한 지역 저장소
- **clone** : 원격 저장소에서 깃허브 프로젝트를 내 PC로 복사



2. 깃과 깃허브 - 깃허브 프로젝트 기여 과정

1. 기여하려는 프로젝트의 저장소를 포크(fork)
2. 나의 PC에 저장소를 클론하기
3. 원격 저장소 Remote 설정하기
4. PR(Pull Request ;병합요청)용 branch 생성하기
5. 오픈소스에서 기여할 부분을 수정
6. PR용 branch에 Push하기
7. 포크해온 저장소에 들어가 PR하기





3. 파이어베이스

1. 파이어베이스 개요
2. 파이어베이스 주요기능
3. 안드로이드 앱 연결
4. 데이터 추가하기

3. 파이어베이스 - 파이어베이스 개요

파이어베이스(FireBase)란?



: 구글에서 제공하는 모바일 및 웹 애플리케이션 개발 플랫폼이다.

인증, 데이터베이스, 푸시알람, 스토리지 등의 백엔드 서비스 플랫폼을 프로젝트 구축 시 자동적으로 만들어 준다. 또한 서버를 구축하기 위해서 리눅스 명령어를 알 필요도 없으며 도메인을 구입할 필요가 없고 개발하는 동안에는 서버를 구입할 필요도 없다. 오랜 시간을 들여 백엔드를 구현할 필요가 없으므로 빠르게 프로토타입을 만들고, 이용자들에게 배포하여 테스트하기 적합하다.

3. 파이어베이스 - 파이어베이스 주요기능



Cloud Firestore

: 글로벌 규모의 모바일 및 웹 앱용 데이터를 쉽게 저장, 동기화, 쿼리할 수 있게 해주는 NoSQL 데이터베이스.



Cloud Storage

: 사진, 동영상 등의 사용자 제작 콘텐츠를 빠르고 손쉽게 저장하고 제공할 수 있도록 설계.



Authentication (인증)

: 안전한 인증 시스템을 손쉽게 구축하도록 지원. 계정(이메일/비밀번호), 전화 인증, 구글, 트위터, 페이스북, 깃허브 로그인 등을 지원하는 엔드 투 엔드 ID 솔루션 제공.



Cloud Messaging

: 서버와 기기를 안정적으로 연결하고, iOS, Android, 웹에서 메시지와 푸시 알림을 무료로 주고받을 수 있음.

3. 파이어베이스 - 파이어베이스 주요기능



Google Analytics

: 최대 500개의 고유 이벤트에 대해 무료로 제한 없는 보고 기능을 제공. 특정 중요 이벤트와 사용자 속성을 SDK가 자동으로 포착. iOS 및 Android 앱의 사용자 행동에 대한 데이터를 파악하여 제품 및 마케팅 최적화와 관련하여 보다 합리적인 결정을 내릴 수 있도록 해줌.

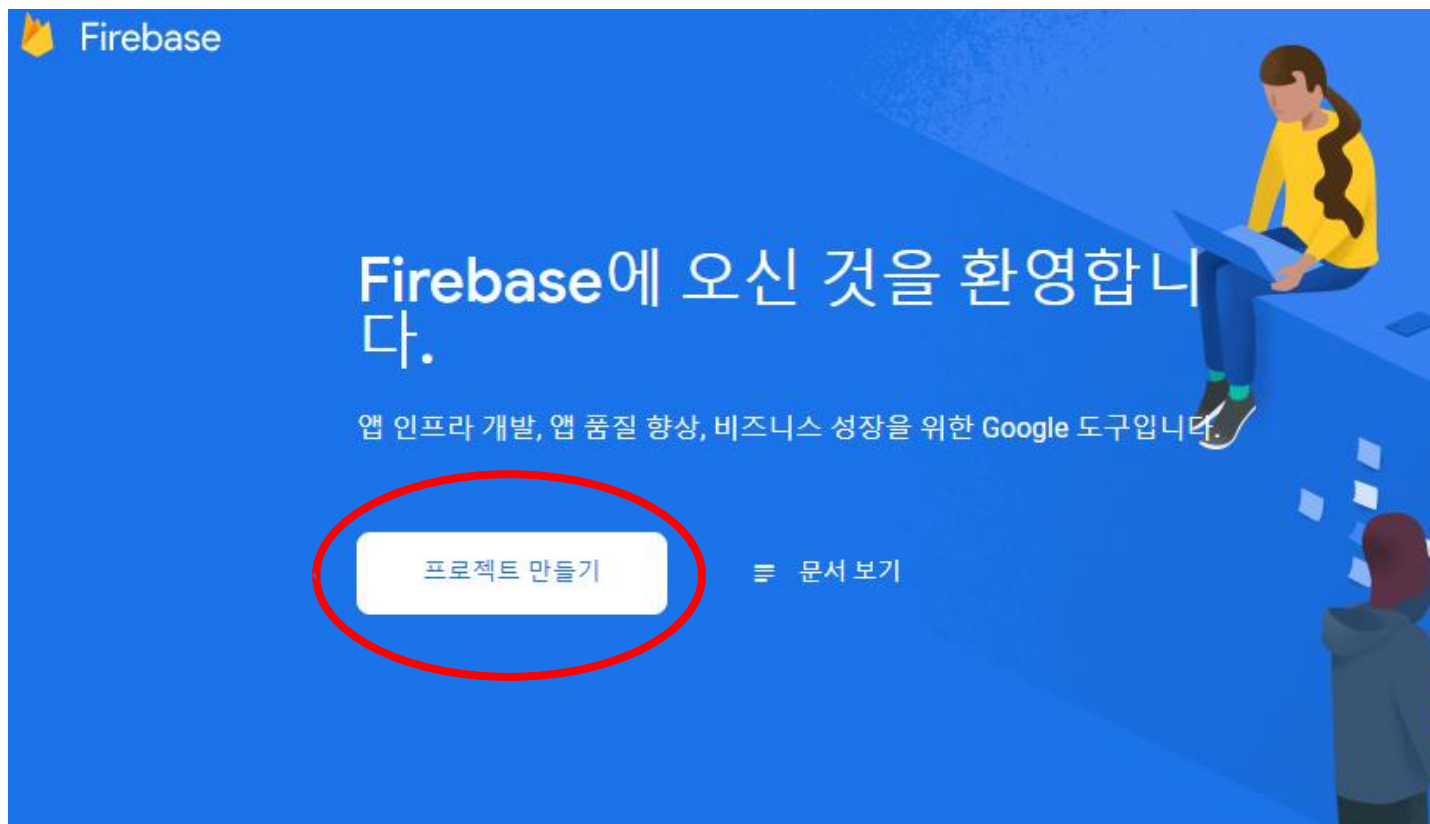


Crashlytics

: 실제 사용자에게 미치는 영향을 기반으로 가장 심각한 비정상 종료의 우선순위를 지정하고 문제를 해결하는 데 도움을 줌. 위치에 관계없이 새로운 오류, 회귀된 오류, 더 높은 비율로 발생하기 시작한 오류에 대한 실시간 경고를 수신할 수 있음.

3. 파이어베이스 - 안드로이드 앱 연결

1. 파이어베이스 로그인 후 프로젝트 만들기 클릭




3. 파이어베이스 - 안드로이드 앱 연결

2. 프로젝트 이름 설정 후 계속 클릭

프로젝트[®] 이름을 지정하여
시작하기

프로젝트 이름

focus on

 focus-on-8585f


계속

3. 파이어베이스 - 안드로이드 앱 연결

3. Google 애널리틱스 관련 설정 후 프로젝트 만들기 클릭

Google 애널리틱스 구성

Google 애널리틱스 계정 선택 또는 만들기 ⓘ

 Default Account for Firebase ▼

이 계정에서 자동으로 새 속성 만들기 ✎

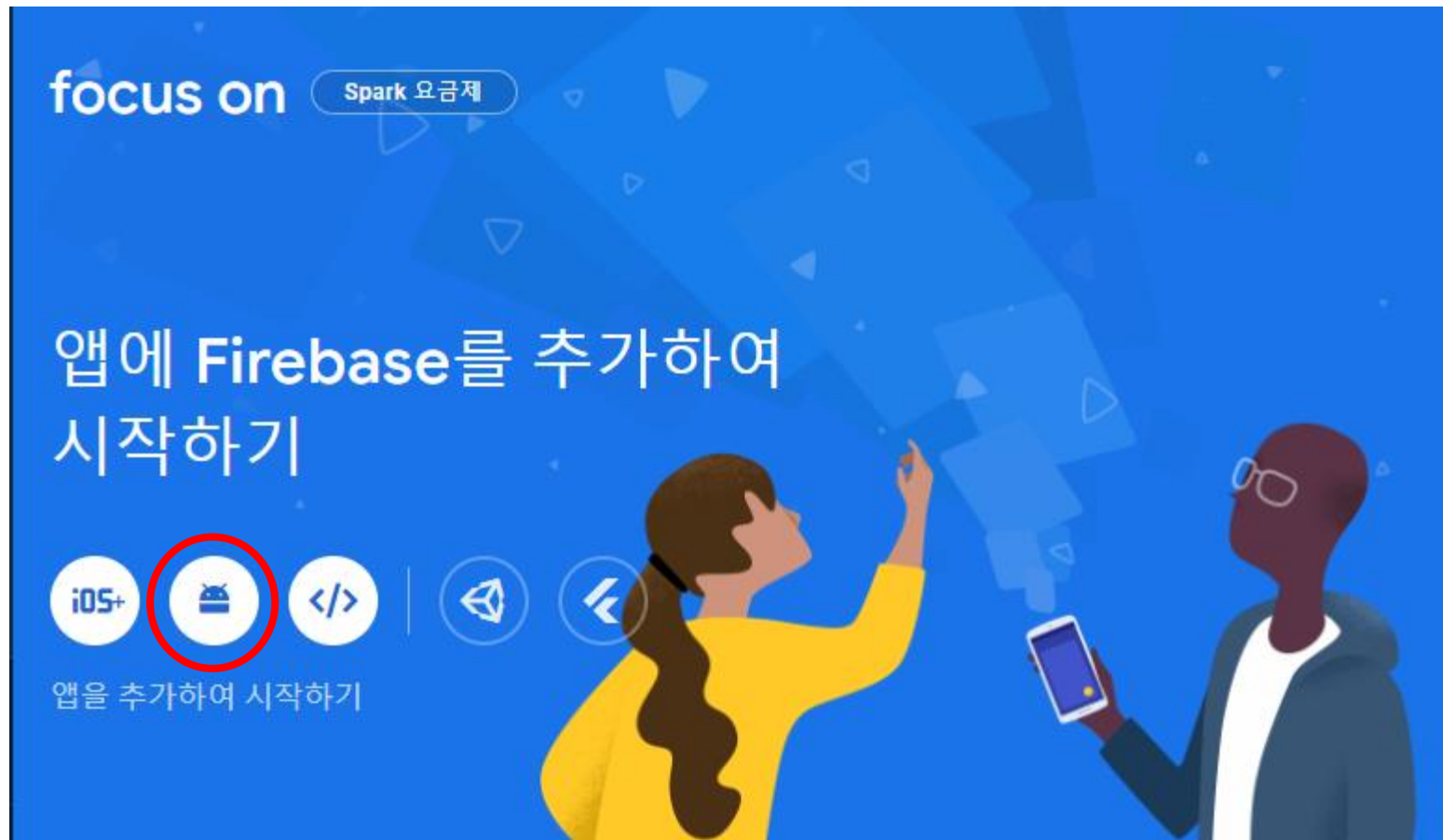
프로젝트를 만들면 선택한 Google 애널리틱스 계정에 새 Google 애널리틱스 속성이 생성되고 Firebase 프로젝트에 연결됩니다. 이 연결을 통해 제품 간에 데이터 흐름이 활성화됩니다. Google 애널리틱스 속성에서 Firebase로 내보낸 데이터에는 Firebase 서비스 약관이 적용되지만 Google 애널리틱스로 가져온 Firebase 데이터에는 Google 애널리틱스 서비스 약관이 적용됩니다. [자세히 알아보기](#)

[이전](#)

프로젝트 만들기

3. 파이어베이스 - 안드로이드 앱 연결

4. 프로젝트 생성 후, 메인화면 중앙에 위치한 안드로이드 아이콘 클릭



3. 파이어베이스 - 안드로이드 앱 연결

5. 안드로이드 프로젝트 패키지, 앱 닉네임, SHA-1을 입력하고 다음 클릭

1 앱 등록

Android 패키지 이름 ①

com.lakue.firebasesample

앱 닉네임 (선택사항) ②

FireBaseSample

디버그 서명 인증서 SHA-1(선택사항) ③

인증서에서 동적 링크, 초대, Google 로그인, 전화번호를 지원하는 데 필요합니다. 설정에서 SHA-1을 수정하세요.

다음

3. 파이어베이스 - 안드로이드 앱 연결

6. google-services.json 파일 다운받고 Project -> app 폴더에 넣기

× Android 앱에 Firebase 추가

1 앱 등록
Android 패키지 이름: net.flow9.thisisKotlin.firebase, 앱 닉네임: 파이어베이스

2 구성 파일 다운로드
Android 스튜디오에 대한 안내(아래 참조) | Unity C++

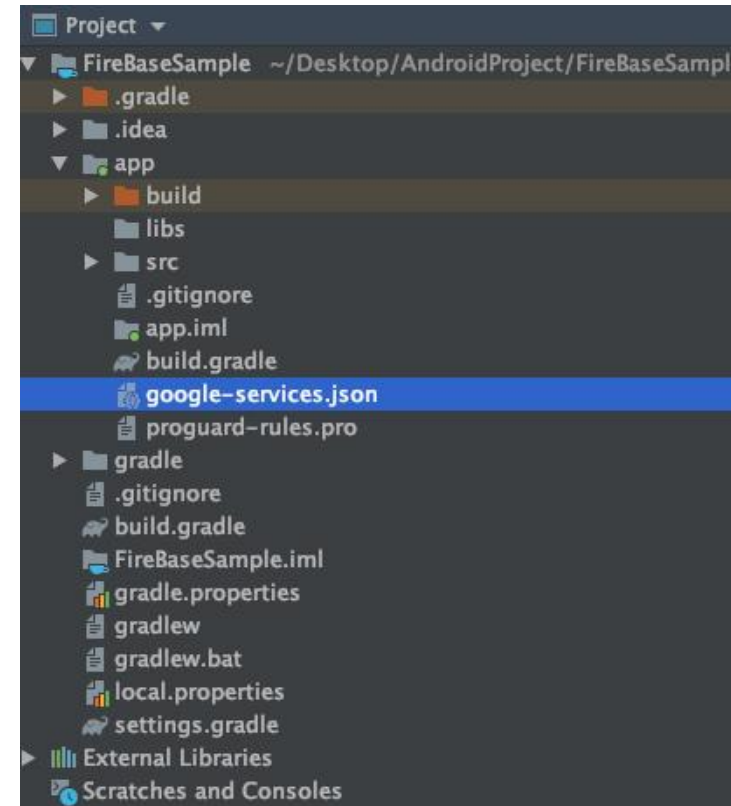
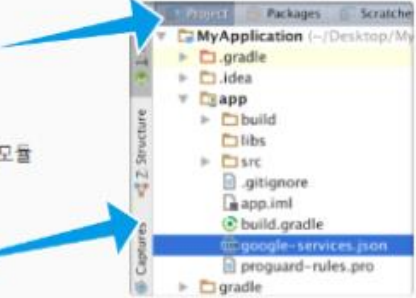
google-services.json 다운로드

Android 스튜디오에서 프로젝트 보기로 전환하여 프로젝트 루트 디렉터리를 표시하세요.

방금 다운로드한 google-services.json 파일을 Android 앱 모듈 루트 디렉터리로 이동하세요.

google-services.json

다음



3. 파이어베이스 - 안드로이드 앱 연결

7. 파이어베이스와 SDK를 연결하기 위해 build.gradle(Project : appname)과 Build.gradle(Module.app)에 아래 코드 입력 후 Sync Now 클릭

```
buildscript {
    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.5.3'
        classpath 'com.google.gms:google-services:4.3.3'
        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

```
apply plugin: 'com.android.application'
apply plugin: 'com.google.gms.google-services'

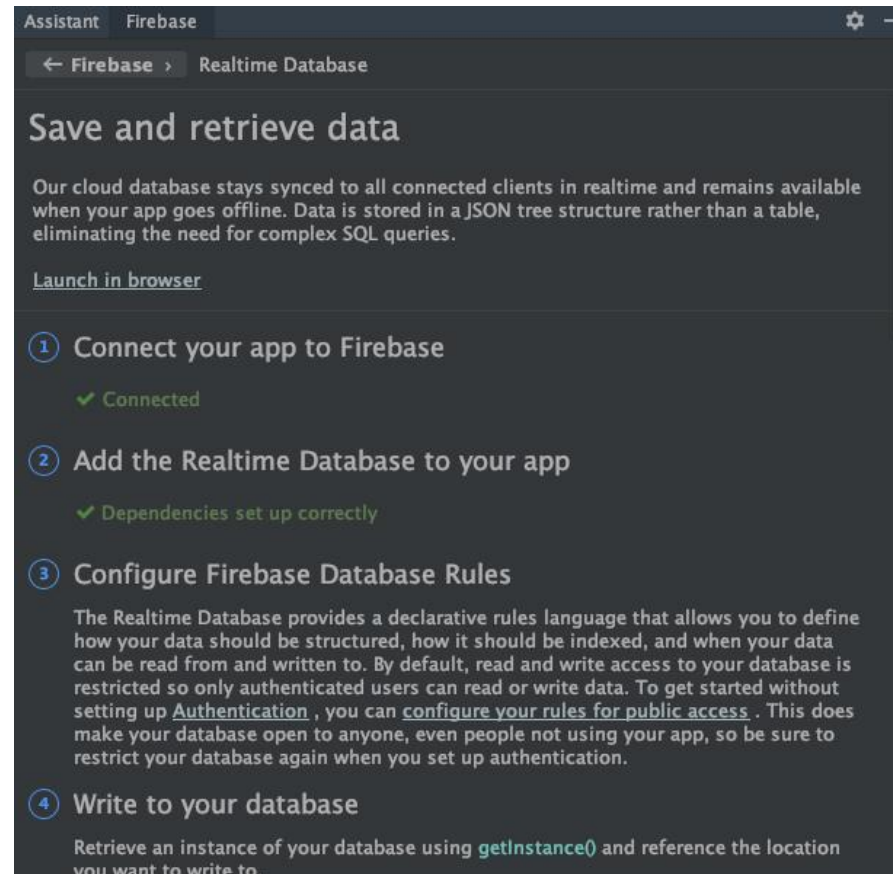
dependencies {
    ...

    implementation 'com.google.firebase:firebase-analytics:17.2.3'
    implementation 'com.google.firebase:firebase-core:17.2.3'
    implementation 'com.google.firebase:firebase-database:19.2.1'
}
```

Gradle files have changed since last project sync. A project sync may be necessary fo... [Sync Now](#)

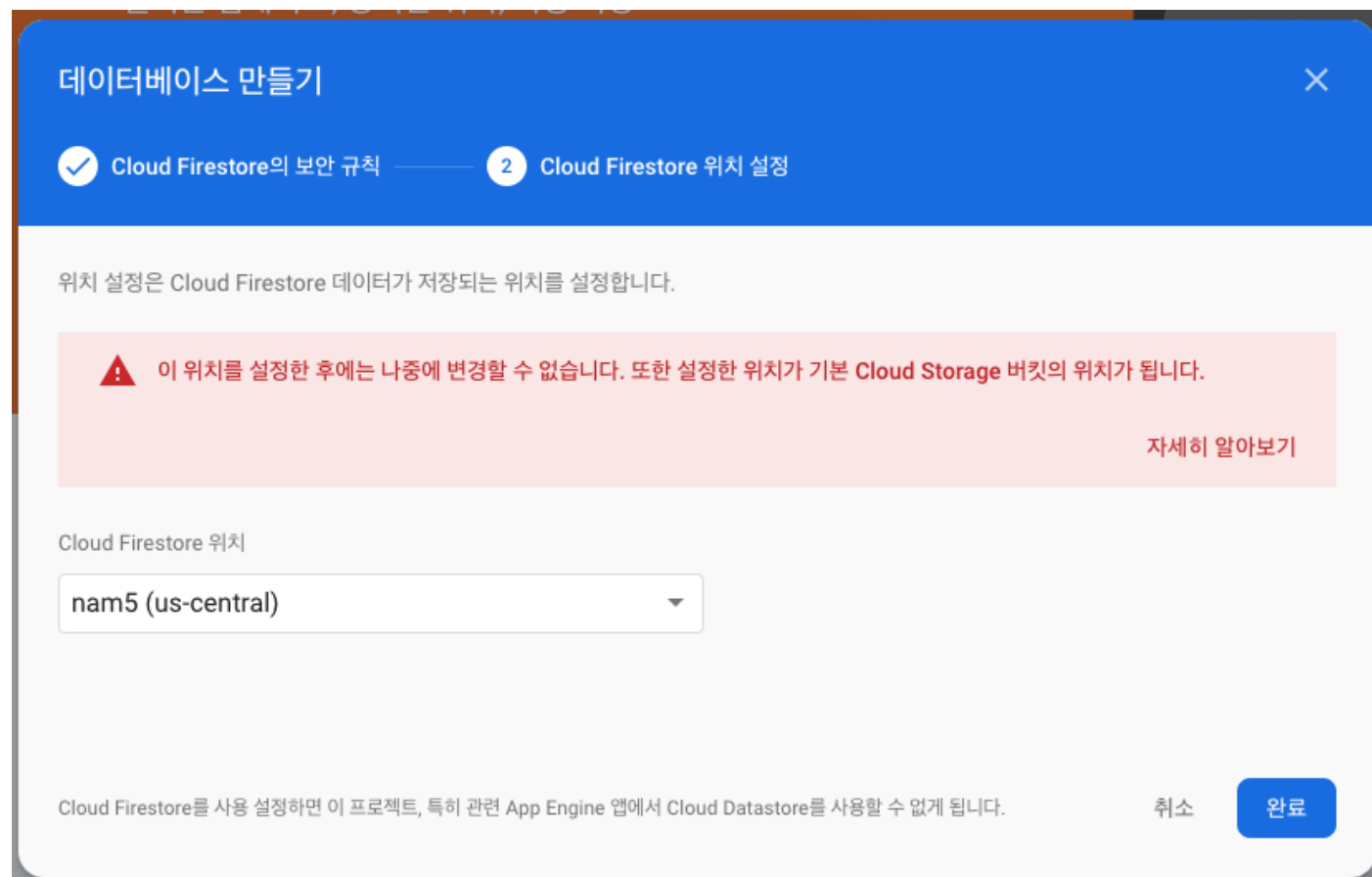
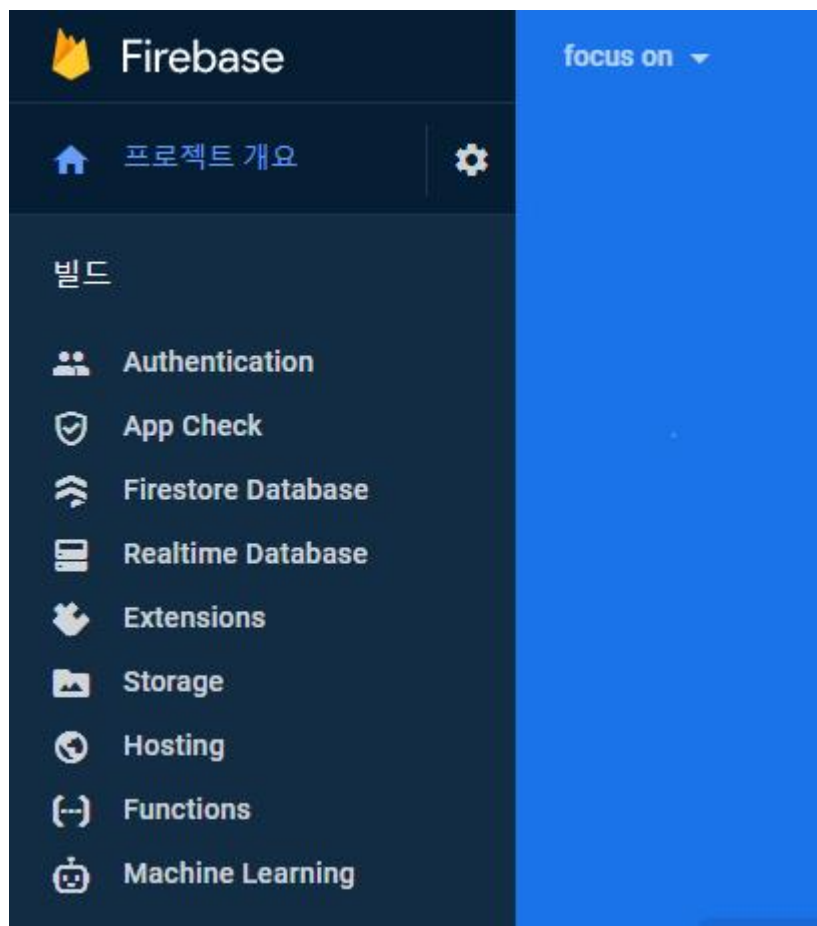
3. 파이어베이스 - 안드로이드 앱 연결

8. Tool > Firebase > Realtime Database > Save and retrieve data 클릭
하여 1번과 2번이 Connected 라고 뜨면 연결 성공



3. 파이어베이스 - 데이터 추가하기

1. 메인화면 좌측에 Firestore Database 클릭하여 데이터베이스 생성



3. 파이어베이스 - 데이터 추가하기

2. 권한 설정을 위해 Realtime Database의 규칙을 다음과 같이 변경

Test ▾

Database

Realtime Database ▾

데이터 규칙 백업 사용량

게시되지 않은 변경사항 | 게시 삭제

시뮬레이터

★ 기본 보안 규칙이 액세스할 수 없도록 잠겨 있습니다.

자세히 알아보기 닫기

1 ▾
2 ▾
3 ▾
4
5
6
7

```
{
  /* Visit https://firebase.google.com/docs/database/security to learn more about security rules. */
  "rules": {
    ".read": true,
    ".write": true
  }
}
```


3. 파이어베이스 - 데이터 추가하기

3. AndroidManifest 에 인터넷 권한을 허용하도록 수정

```
<!-- 인터넷 사용 권한 -->  
<uses-permission android:name="android.permission.INTERNET" />
```

3. 파이어베이스 - 데이터 추가하기

4. User.java 클래스 생성 후 아래와 같이 코드 작성

```
private DatabaseReference mDatabase;// ...
mDatabase = FirebaseDatabase.getInstance().getReference();

@IgnoreExtraProperties
public class User {

    public String userName;
    public String email;

    public User() {
        // Default constructor required for calls to DataSnapshot.getValue(User.class)
    }

    public User(String userName, String email) {
        this.userName = userName;
        this.email = email;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    @Override
    public String toString() {
        return "User{" +
            "userName='" + userName + '\'' +
            ", email='" + email + '\'' +
            '}';
    }
}
```

```
private void writeNewUser(String userId, String name, String email) {
    User user = new User(name, email);

    mDatabase.child("users").child(userId).setValue(user)
        .addOnSuccessListener(new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {
                // Write was successful!
                Toast.makeText(MainActivity.this, "저장을 완료했습니다.", Toast.LENGTH_SHORT).show()
            }
        })
        .addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                // Write failed
                Toast.makeText(MainActivity.this, "저장을 실패했습니다.", Toast.LENGTH_SHORT).show()
            }
        });
}

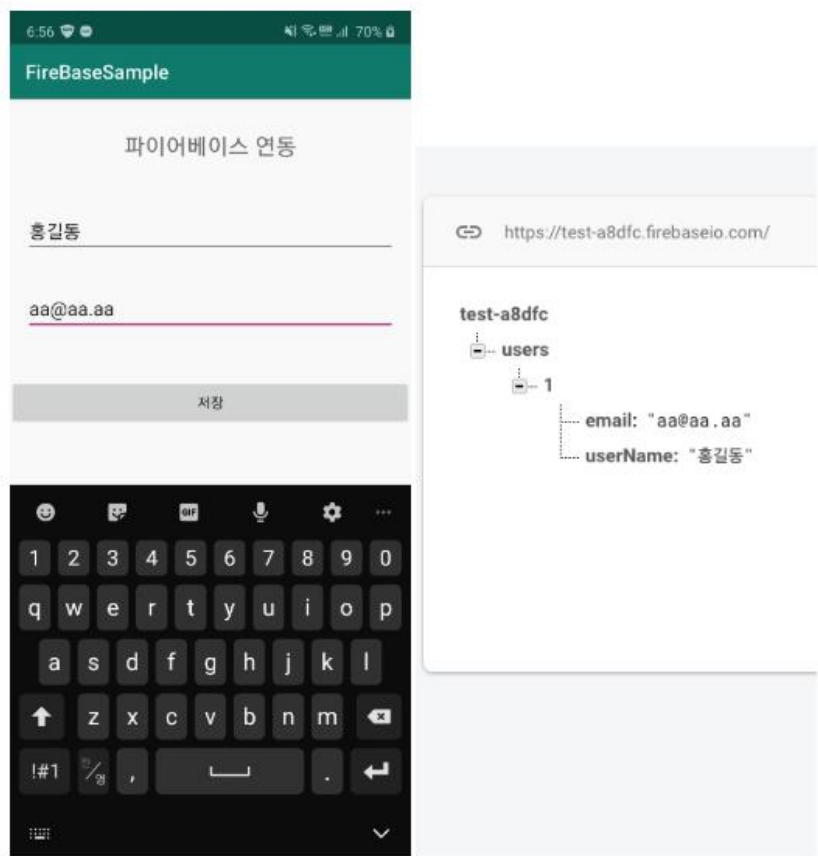
btn_save.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String getUserName = et_user_name.getText().toString();
        String getUserEmail = et_user_email.getText().toString();

        //hashmap 만들기
        HashMap result = new HashMap<>();
        result.put("name", getUserName);
        result.put("email", getUserEmail);

        writeNewUser("1",getUserName,getUserEmail);
    }
});
```

3. 파이어베이스 - 데이터 추가하기

5. 데이터베이스 안에 데이터가 추가 되는지 확인



감사합니다