PREFERENCE 활용하기

기초 원리 학습

OneKick

- 쉐어드 프리퍼런스 저장/불러오기 (Shared Preference) 기능을 구현해 보자.
- 앱에서 데이터 저장 방법
  - sqlite(DataBase)를 이용
  - 쉐어드 프리퍼런스(Shared Preference)를 이용
    - sqlite(DataBase)를 사용하지 않아도 데이터 저장이 가능
    - sqlite의 사용보다 좀 더 쉽게 사용할 수 있지만 하지만 대용량의 데이터일 때 sqlite보다 느린 단점
- 앱이 종료 후 다시 실행되도 저장한 데이터는 삭제되지 않으나 앱을 삭제시에는 데이터도 삭제

기초 원리 학습

OneKick

- SharedPreference를 이용해 저장하기 위해서는 먼저 SharedPreference 를 선언
  - public final String PREFERENCE = "com.studio572.samplesharepreference";
  - SharedPreferences pref = getSharedPreferences(PREFERENCE, MODE_PRIVATE);
    - // SharedPreferences 의 데이터를 저장/편집 하기위해 Editor 변수를 선언
  - SharedPreferences.Editor editor = pref.edit();
    - // key값에 value값을 저장한다.
    - // String, boolean, int, float, long 값 모두 저장가능
  - editor.putString(key, value);
    - // 메모리에 있는 데이터를 저장장치에 저장한다.
  - editor.commit();
  - SharedPreferences pref = getSharedPreferences(PREFERENCE, MODE_PRIVATE);

기초 원리 학습

OneKick

- 첫 번째 매개변수(PREFERENCE)
  - 저장/불러오기 하기 위한 key

- 이 고유키로 앱의 할당된 저장소에 XML 파일로 저장
  - 저장소: data/data/[패키지 이름]/shared_prefs
  - 파일명: "com.studio572.samplesharepreference.xml"

- 이 때 xml 파일명은 사용자 정의가 가능

- 두번째 매개변수(MODE_PRIVATE) : 프리퍼런스의 저장 모드 정의

  - [MODE_PRIVATE : 이 앱 안에서 데이터 공유]
  - [MODE_WORLD_READABLE : 다른 앱과 데이터 읽기 공유]
  - [MODE_WORLD_WRITEABLE : 다른 앱과 데이터 쓰기 공유]

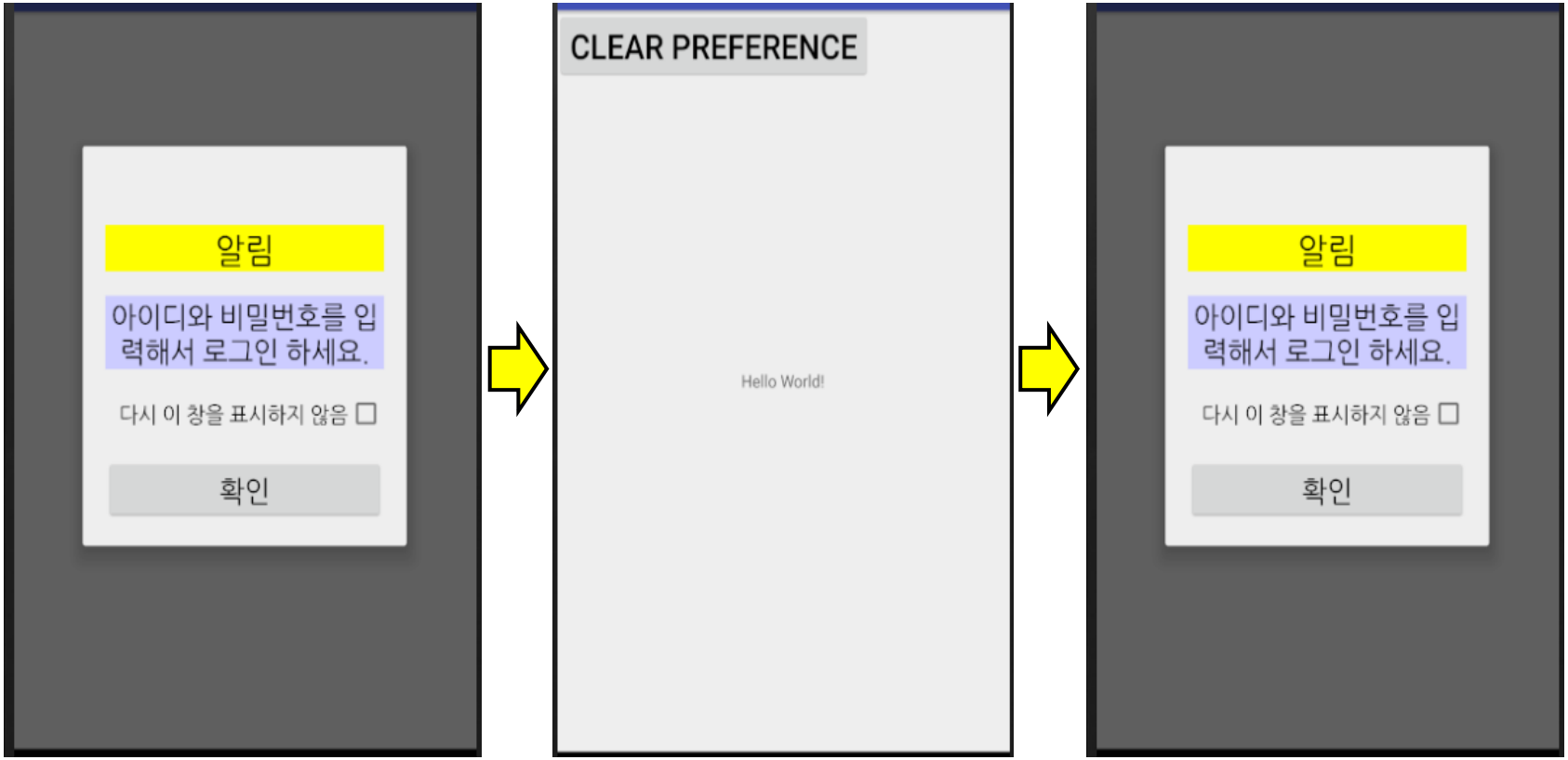
  - 저장된 데이터는 아래와 같이 불러올 수 있다. 먼저 SharedPreference 를 선언.
  - 저장했을 때와 같은 key로 xml에 접근.

  - SharedPreferences pref = getSharedPreferences(PREFERENCE, MODE_PRIVATE);
  - key에 해당한 value를 불러온다.

  - 두번째 매개변수는 , key에 해당하는 value값이 없을 때에는 이 값으로 대체.
  - String result = pref.getString(key, "");

구현 목표

- 처음 실행 시 알림 창 표시
  - 체크하면 다시 표시하지 않음
  - 프리퍼런스를 초기화 하면 다시 알림창이 표시됨



기초 실습 activity_main.xml 파일 -1/2

OneKick

- 화면을 터치 했을 때 다이얼로그를 표시하는 기능을 구현

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

기초 실습

OneKick

- activity_main.xml 파일 -2/2

```
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Clear  Preference"
        android:textSize="28dp"
        android:textColor="#000000"
        />

</android.support.constraint.ConstraintLayout>
```

notification.xml 1/4

OneKick

- 팝업창은 일종의 공지사항
  - 다시 보고 싶지 않을 때 체크하면 그것을 기억했다가 다시 표시하지 않게 하는 기능

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=" http://schemas.android.com/apk/res/android"

    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="match_parent" // 부모를 안벗어나게 최대화
    android:orientation="vertical"
    android:gravity="center"
    android:padding="10dp"
    >
    <TextView
        android:layout_width="match_parent" // 부모를 안벗어나게 최대화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:text="알림"
        android:textColor="#000000"
        android:textSize="28dp"
        android:background="#FFFF00"
        android:gravity="center"
        android:layout_margin="10dp"
        />
```

- notification.xml 이며 팝업창에서 표시할 레이아웃 -2/4

```
<TextView
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
    android:text="아이디와 비밀번호를 입력해서 로그인 하세요."
    android:textColor="#000000"
    android:textSize="24dp"
    android:background="#CCCCFF"
    android:gravity="center"
    android:layout_margin="10dp"
    />

<LinearLayout
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
    android:gravity="right|center"
    android:layout_margin="10dp"
    >
```

- notification.xml 이며 팝업창에서 표시할 레이아웃 -3/4

```
<TextView
        android:layout_width="wrap_content" // 컨텐츠에 맞게 최소화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:text="다시 이 창을 표시하지 않음"
        android:textColor="#000000"
        android:textSize="18dp"
        android:gravity="right|center"
        />

<CheckBox
        android:id="@+id/checkBox"
        android:layout_width="wrap_content" // 컨텐츠에 맞게 최소화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화 />

</LinearLayout>
```

기초 실습

OneKick

- notification.xml 이며 팝업창에서 표시할 레이아웃 -4/4

```
        <Button
            android:id="@+id/button"
            android:layout_width="match_parent" // 부모를 안벗어나게 최대화
            android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
            android:text="확인"
            android:textColor="#000000"
            android:textSize="24dp"
            android:gravity="center"
            android:layout_margin="10dp"
            />
</LinearLayout>
```

MainActivity.java 1/4

```
package com.iot.preferencetest;
import android.app.Dialog;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity; // 액티비티 부모 클래스
import android.os.Bundle; // 액티비티 생성 번들
import android.util.Log; // 로그 출력 목적
import android.view.View; // 뷰계열 최상위 클래스
import android.widget.Button; // 버튼 뷰 사용
import android.widget.CheckBox;

public class MainActivity extends AppCompatActivity { // 메인 화면
        private static final String TAG = "MainActivity";
        private static final String PREFERENCE_NAME = "MyPreferecne";
        private static final String KEY_CHECK = "KeyCheck";
        private Dialog dialog;
        private CheckBox checkBox;
```

MainActivity.java 2/4

OneKick

```
@Override // 부모 메소드 재정의
protected void onCreate(Bundle savedInstanceState) { // 화면생성 이벤트
    super.onCreate(savedInstanceState); // 부모 생성자 호출
    setContentView(R.layout.activity_main); // 메인 화면 표시
    boolean isChecked = loadPreference(KEY_CHECK);
    if(!isChecked) {
        dialog = new Dialog(this);
        dialog.setContentView(R.layout.notification);
        dialog.show();
        checkBox = dialog.findViewById(R.id.checkBox);
        Button button = dialog.findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() { // 클릭리스
        터 생성
            @Override // 부모 메소드 재정의
            public void onClick(View v) { // 클릭 이벤트 처리
            dialog.dismiss();
            boolean isChecked = checkBox.isChecked();
            savePreference(KEY_CHECK, isChecked);
        }
    });
}
```

MainActivity.java 3/4

OneKick

```
Button button = (Button)findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() { // 클릭리스터 생성
        @Override // 부모 메소드 재정의
        public void onClick(View v) { // 클릭 이벤트 처리
                clearPreference();
                }
                });
        return;
}
private void clearPreference() {
        SharedPreferences preferences =
        getSharedPreferences(PREFERENCE_NAME, MODE_PRIVATE);
        SharedPreferences.Editor editor = preferences.edit();
        editor.clear();
        editor.commit();
}
private boolean loadPreference(String key) {
        SharedPreferences preferences =
        getSharedPreferences(PREFERENCE_NAME, MODE_PRIVATE);
        return preferences.getBoolean(key, false);
}
```

MainActivity.java 4/4

OneKick

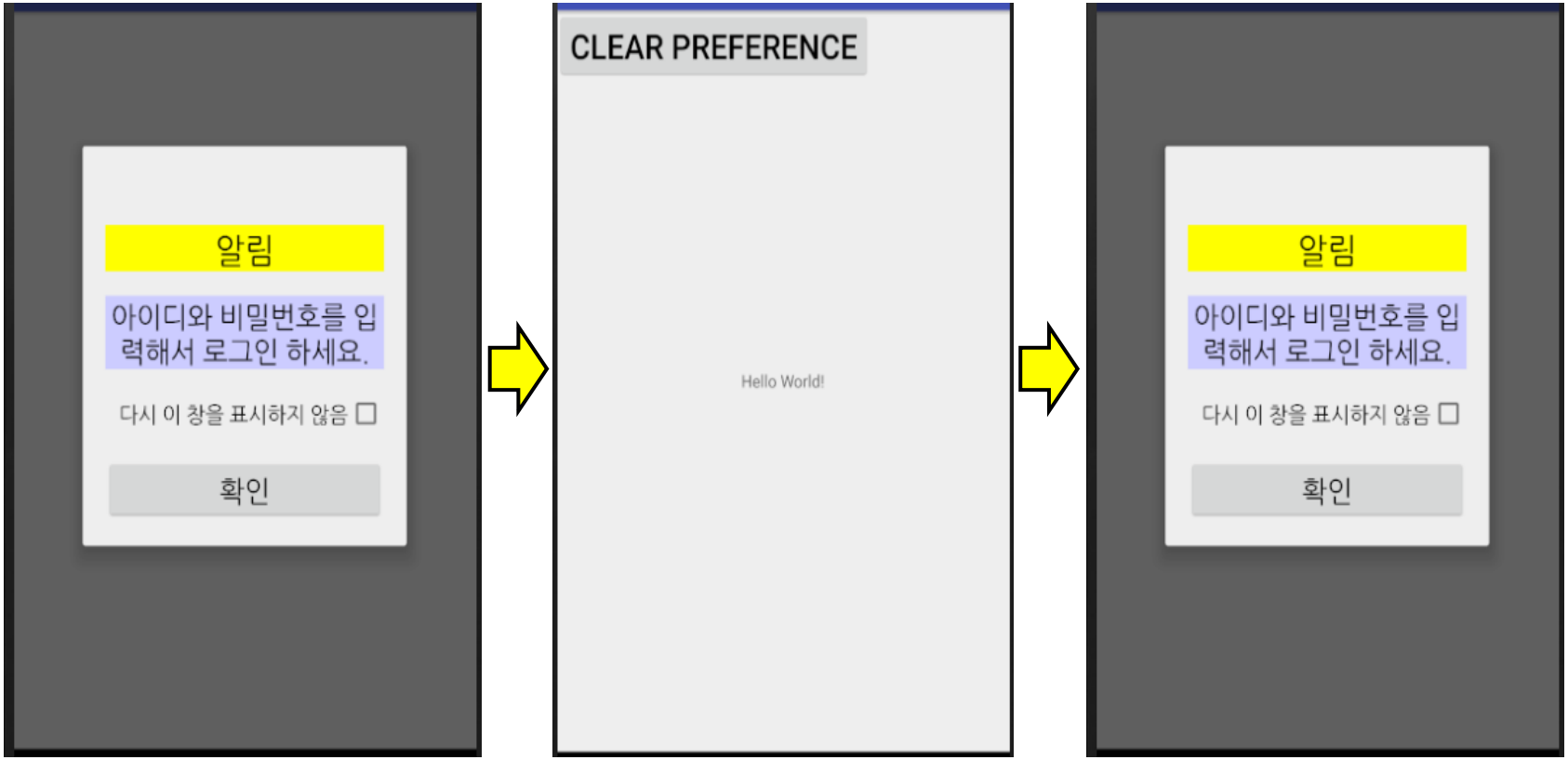
```
    private void savePreference(String key, boolean isChecked) {
        SharedPreferences preferences =
        getSharedPreferences(PREFERENCE_NAME, MODE_PRIVATE);
        SharedPreferences.Editor editor= preferences.edit();
        editor.putBoolean(key, isChecked);
        editor.commit();
    }

    @Override // 부모 메소드 재정의
    protected void onPause() {
        super.onPause();
        Log.i(TAG,"onPause");
    }

    @Override // 부모 메소드 재정의
        protected void onStop() {
        super.onStop();
        Log.i(TAG,"onStop");
    }
}
```

동작 테스트

- 처음 실행 시 알림 창 표시
  - 체크하면 다시 표시하지 않음
  - 프리퍼런스를 초기화 하면 다시 알림창이 표시됨



심화 학습

OneKick

- 저장된 데이터의 초기화
  - 이전에 기록해 두었던 SharedPreference 저장 기록을 삭제
    - 이렇게 삭제해 주지 않으면 다시는 팝업창을 볼 수 없기 때문
  - Settings에 들어가서 애플리케이션을 삭제하고 다시 설치
    - 또는 데이터 삭제 기능을 이용

- Preference를 이용하면 사용자 경험을 토대로한 서비스를 제공 가능
  - 최근 사용한 ID를 기록하게 하거나 최근 접속한 홈페이지의 URL을 기억하게 할 수도 있음.
  - 사용자가 최근에 보던 페이지를 바로 표시하고 싶을때 Preference에 저장

심화 학습

OneKick

- Preference는 로컬 폰에 저장
  - 중요하지 않은 사용자 편의를 위한 정보를 저장하기 위한 목적으로 주로 사용.
  - 데이터 량이 많지 않고 가끔 사용되는 목적으로는 적당

- 대량의 센싱 데이터나 고객데이터 등을 저장하는 경우
  - SQLite와 같은 DBMS를 사용하는 것이 속도나 자원 활용 면에서 훨씬 유리