모바일 프로그래밍

07 파일 처리 1

2017 2학기

강승우

Computer Science and Engineering, KOREATECH

안드로이드에서 데이터 저장하기

- 애플리케이션에서 생성한 데이터를 저장하기 위한 방법
  - 파일 이용
    - 내부 저장: 내장 메모리
    - 외부 저장: 외장 메모리 (SD 카드)
  - 공유 프레퍼런스 (Shared Preferences) 이용
    - 애플리케이션 환경 설정 정보
  - 데이터베이스 이용
    - SQLite 데이터베이스
  - 서버 이용
    - 네트워크를 통한 데이터 전송

안드로이드 파일 처리

- 내부 저장 공간(내장 메모리)을 이용한 파일 입출력
  - 내부 저장 공간은 항상 이용 가능
  - 해당 애플리케이션만 접근이 가능한 파일 생성 가능
    - 애플리케이션을 제거하면 생성된 파일도 함께 삭제됨

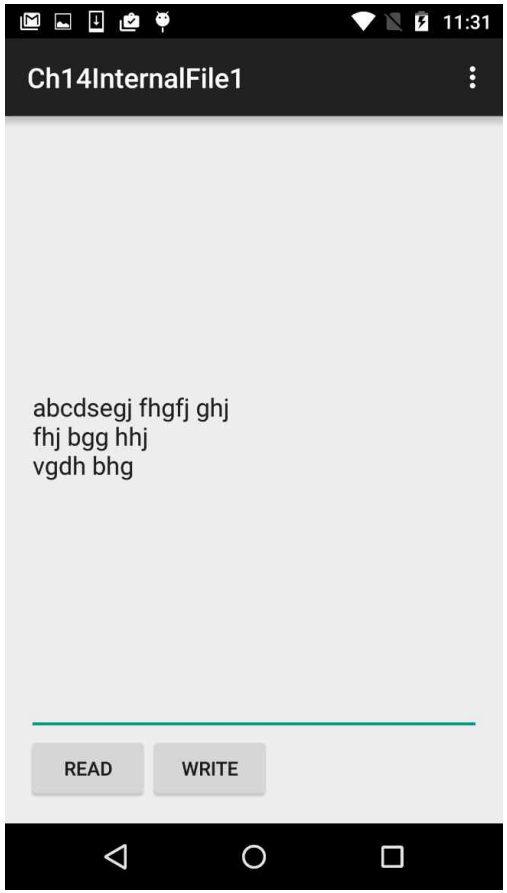
- 외부 저장 공간을 이용한 파일 입출력
  - 스마트폰에 탈착이 가능한 SD 카드를 사용하는 경우 항상 이용 가능한 것은 아님
  - 외장 메모리의 공용 디렉토리에 저장된 파일들은 다른 애플리케이션에서도 접근 가능하고, 사용자에 의해서 변경될 수 있음
    - 사용자가 애플리케이션을 제거하더라도 공용 디렉토리에 저장된 파일은 삭제되지 않음
  - 외장 메모리를 이용하는 경우에도 특정 애플리케이션이 사적으로 사용하는 파일 생성 가능 (애플리케이션 제거 시 삭제)

내장 메모리 파일 입출력

- Context class에 정의된 파일 입출력 메소드 이용
  - FileOutputStream openFileOutput(String name, int mode)
  - FileInputStream openFileInput(String name)
  - boolean deleteFile(String name)

- Mode
  - Context.MODE_PRIVATE
    - 다른 애플리케이션 패키지에서 접근할 수 없는 파일을 생성할 때 사용
  - Context.MODE_APPEND
    - 기존 파일의 끝에 데이터를 추가하기 위한 용도로 파일을 열 때 사용
  - Context.MODE_WORLD_READABLE / Context.MODE_WORLD_WRITABLE
    - 다른 애플리케이션이 접근 가능하도록 하는 모드.
    - 보안 상의 이유로 deprecated. 사용하지 않는 것이 좋다

텍스트 파일 입출력 예제



- 예제 프로젝트 이름: Ch14InternalFile1

- EditText에 텍스트 입력

- WRITE 버튼을 누르면 지정된 파일 이름을 갖는 파일에 저장
  - String FILENAME = “text.txt”;
  - 예제는 위와 같이 파일 이름을 하드 코딩한 상태

- READ 버튼을 누르면 동일한 이름의 파일을 읽어서 그 내용을 EditText에 설정

파일 쓰기

```
Button writeBtn = (Button)findViewById(R.id.write);
writeBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {
            FileOutputStream fos = openFileOutput(FILENAME, Context.MODE_PRIVATE);
            fos.write(edit.getText().toString().getBytes());
            fos.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
});
```

- Private 모드로 파일 생성 후 쓰기
- 쓰기 후 close

File IO 관련 API를 사용하는 경우 IOException 처리를 해주어야 함

파일 읽기

- 파일 읽기 위해 파일 오픈
- 버퍼 생성 후 읽기 수행

- 읽은 내용을 화면에 표시하기 위해 EditText 객체에 설정

```
Button readBtn = (Button)findViewById(R.id.read);
readBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {
            FileInputStream fis = openFileInput(FILENAME);
            byte[] buffer = new byte[fis.available()];
            fis.read(buffer);

            edit.setText(new String(buffer));
            fis.close();

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
});
```

파일 경로

- 안드로이드 내장 메모리 공간
  - 리눅스 파일시스템
    - 안드로이드 시스템 관련한 모든 파일
    - 사용자가 설치한 앱 파일
  - 앱이 설치되면 그 앱에서 사용할 수 있는 별도의 공간이 할당됨

  → 앱 홈 폴더: /data/data/패키지명

- /data/data/패키지명/files
  - openFileInput, openFileOutput, deleteFile 등의 메소드에서 접근하는 폴더
  - Context 클래스에 정의된 getFilesDir() 메소드를 이용해 경로를 얻어올 수 있음

관련 메소드

- FILE getFilesDir()
- String[] fileList()
  - 애플리케이션이 현재 저장한 파일 리스트 반환
- FILE getFileStreamPath(String name)
  - 특정 파일의 경로를 얻어옴
  - /data/data/files/파일명

텍스트 파일 입출력 예제 2

- 예제 프로젝트 이름: Ch14InternalFile2
  - TextFileManager라는 별도의 클래스로 파일 쓰기, 읽기, 삭제를 처리하는 예제
- MainActivity에서는 TextFileManager 객체를 생성하여 이용
- 불러오기, 저장, 삭제 버튼을 선택하였을 때 TextFileManager 객체의 관련 메소드 호출하여 필요한 작업 수행
  - save(), load(), delete() 메소드
- 예제 소스 참고