Embedded System Practice Lab 11

2016311821 한승하

<Exercise 1>

```
public class Math{
        public native int sum(int x, int y);
        public native int difference(int x, int y);
        public native int product(int x, int y);
        public native double average(int x, int y);
}
```

Exercise를 위해 Math class를 선언하고 4개의 method를 만들어 주었습니다.

```
public class Main{
        public static void main(String[] args){
                System.loadLibrary("Math");
                Math Math_unit = new Math();
                Math_unit.sum(10, 1);
                Math_unit.difference(1, 6);
                Math_unit.product(3,7);
                Math_unit.average(10, 20);
        }
}
```

위 Method를 Test할 Main문 또한 작성해 주었습니다.

```c
/* DO NOT EDIT THIS FILE - it is machine generated */
#include <jni.h>
/* Header for class Math */

#ifndef _Included_Math
#define _Included_Math
#ifdef __cplusplus
extern "C" {
#endif
/*
 * Class:     Math
 * Method:    sum
 * Signature: (II)I
 */
JNIEXPORT jint JNICALL Java_Math_sum
  (JNIEnv *, jobject, jint, jint);

/*
 * Class:     Math
 * Method:    difference
 * Signature: (II)I
 */
JNIEXPORT jint JNICALL Java_Math_difference
  (JNIEnv *, jobject, jint, jint);

/*
 * Class:     Math
 * Method:    product
 * Signature: (II)I
 */
JNIEXPORT jint JNICALL Java_Math_product
  (JNIEnv *, jobject, jint, jint);

/*
 * Class:     Math
 * Method:    average
 * Signature: (II)D
 */
JNIEXPORT jdouble JNICALL Java_Math_average
  (JNIEnv *, jobject, jint, jint);

#ifdef __cplusplus
}
#endif
#endif
~
~
```

이를 컴파일하여 다음과 같은 헤더파일이 만들어졌습니다.

```c
#include <stdio.h>
#include "Math.h"

JNIEXPORT jint JNICALL Java_Math_sum(JNIEnv *env, jobject obj, jint x, jint y){
        printf("Sum is: %d\n",(x+y));
        return x+y;
}

JNIEXPORT jint JNICALL Java_Math_difference(JNIEnv *env, jobject obj, jint x, jint y){
        printf("difference is: %d\n",(x-y));
        return x-y;
}

JNIEXPORT jint JNICALL Java_Math_product(JNIEnv *env, jobject obj, jint x, jint y){
        printf("product is: %d\n",(x*y));
        return x*y;
}

JNIEXPORT jdouble JNICALL Java_Math_average(JNIEnv *env, jobject obj, jint x, jint y){
        printf("Average is: %d\n",(x+y)/2);
        return (x+y)/2;
}
~
~
```

이제 이를 이용해 C코드를 작성하였습니다.

```
Sum is: 11
difference is: -5
product is: 21
Average is: 15
```

위와 같이 정상적으로 실행되었습니다.

<Exercise 2>

activity_main.xml | native-lib.cpp | MainActivity.java

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <EditText
            android:id="@+id/inputnumber"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:inputType="numberSigned|numberDecimal"
            />

        <Button
            android:id="@+id/button"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:onClick="onClick"
            android:text="Display" />

    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise2를 위해 새로운 project를 만들어주고 Layout을 잡아주었습니다.

```java
package com.vogella.android.service.myapplication;

import androidx.appcompat.app.AppCompatActivity;
import android.view.View;
import android.os.Bundle;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    private EditText text;
    // Used to load the 'native-lib' library on application startup.
    static {
        System.loadLibrary( libname: "native-lib");
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Example of a call to a native method
        text = (EditText) findViewById(R.id.inputnumber);
    }
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.button:
                Integer inputnumber = Integer.parseInt(text.getText().toString());
                printtext(inputnumber);
                break;
        }
    }

    /**
     * A native method that is implemented by the 'native-lib' native library,
     * which is packaged with this application.
     */
    public native void printtext(int input);
}
```

위와 같이 MainActivity를 수정해 주었습니다. Display버튼을 누르면 text에 쓰여 있는 값을 읽어 Integer형태로 printtext() native함수로 넘겨주었습니다.

activity_main.xml | native-lib.cpp | MainActivity.java

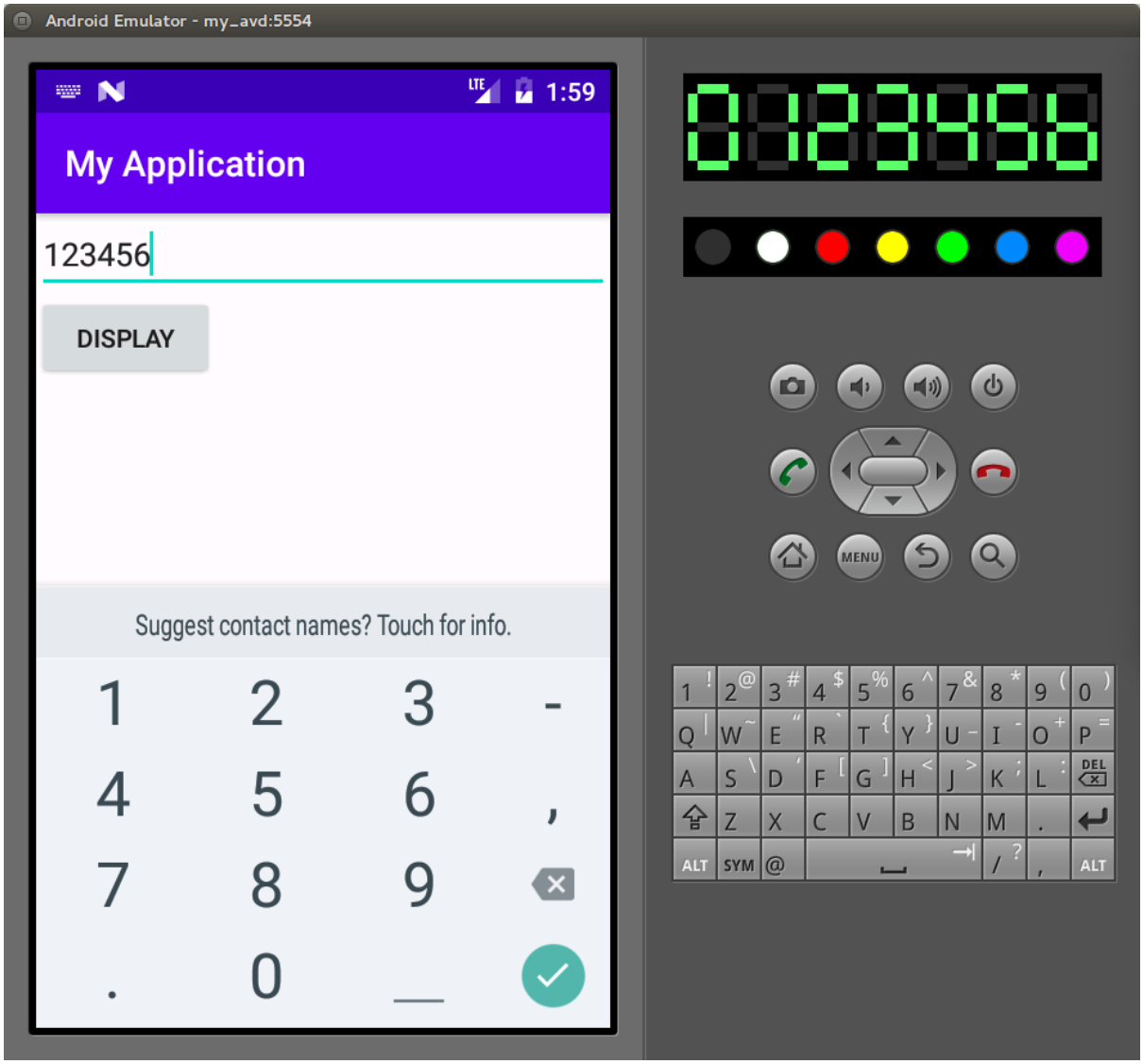
```cpp
#include <jni.h>
#include <string>
#include <unistd.h>

extern "C" JNIEXPORT void JNICALL
MainActivity.printtext(
        JNIEnv* env,
        jobject MainActivity /* this */
        ,jint input) {
    syscall(322,input);
}
```

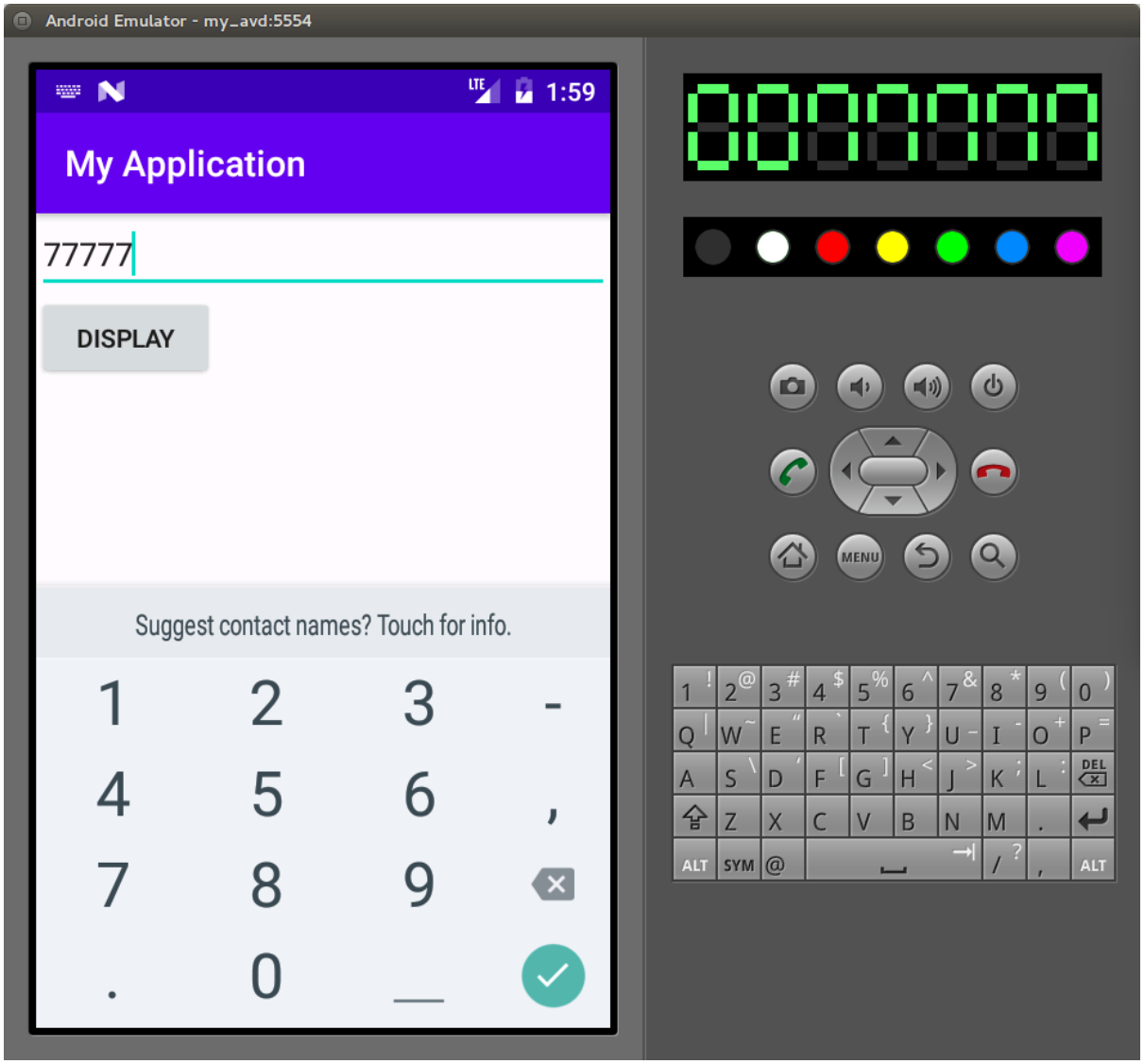
위는 Printtext함수입니다. 받은 input을 system call로 넘겨주게 됩니다.

```cpp
asmlinkage long sys_7seg_control(int num)
{
        if(num>9999999)
        {
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT0, 0x77);
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT1, 0x77);
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT2, 0x77);
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT3, 0x77);
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT4, 0x77);
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT5, 0x77);
                GOLDFISH_SEGMENT_WRITE(s_data, SEGMENT6, 0x77);
                return 0;
        }
        int start_num = num;
        int temp;
        int idx;
        for(idx=6;idx>=0;idx--)
        {
                temp = start_num%10;
                start_num = start_num/10;
                switch(temp){
                        case 0: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x77);
                                break;
                        case 1: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x24);
                                break;
                        case 2: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x5d);
                                break;
                        case 3: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x6d);
                                break;
                        case 4: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x2e);
                                break;
                        case 5: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x6b);
                                break;
                        case 6: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x7a);
                                break;
                        case 7: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x27);
                                break;
                        case 8: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x7f);
                                break;
                        case 9: GOLDFISH_SEGMENT_WRITE(s_data, idx*0x4, 0x2f);
                                break;
                }
        }
        return 0;
}
```

이때 system call 322번은 이전에 작성했던 sys_7segment_contorl syscall을 사용하였습니다.

아래는 완성된 어플리케이션의 testing입니다.




Android Emulator - my_avd:5554
1:59
My Application
77777
DISPLAY
Suggest contact names? Touch for info.
1 2 3 -
4 5 6 ,
7 8 9
. 0 _
MENU