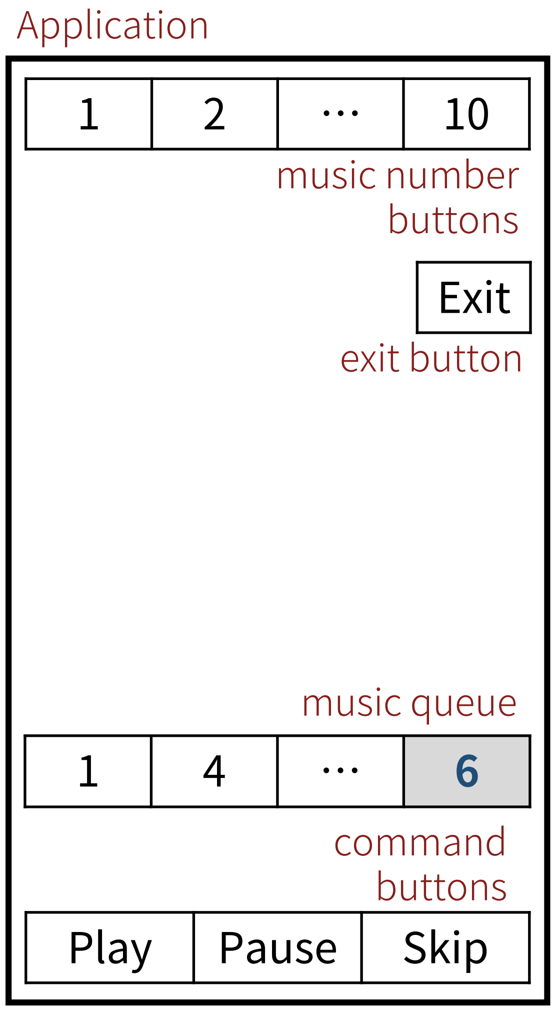
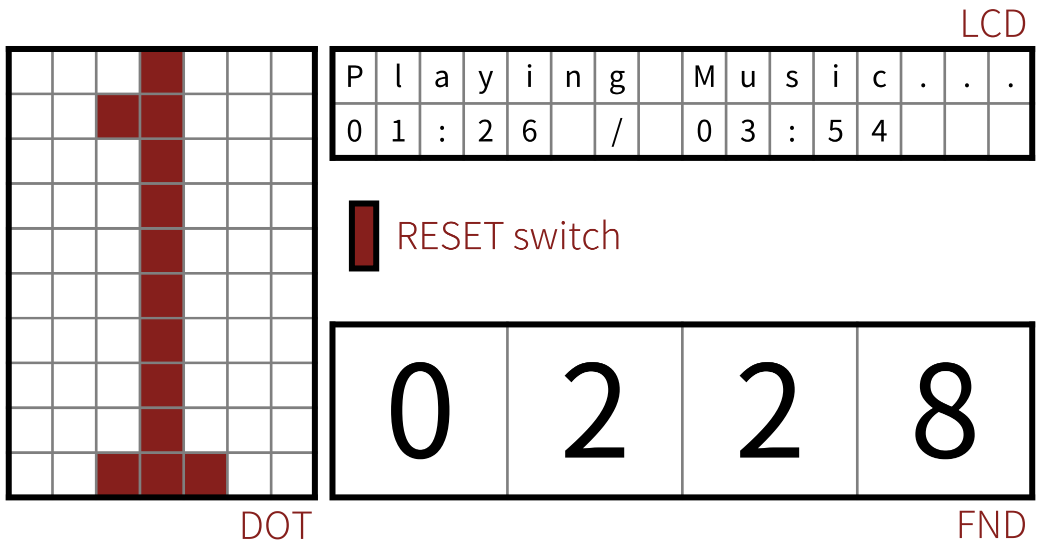
Music Playback Application and Recommendation System

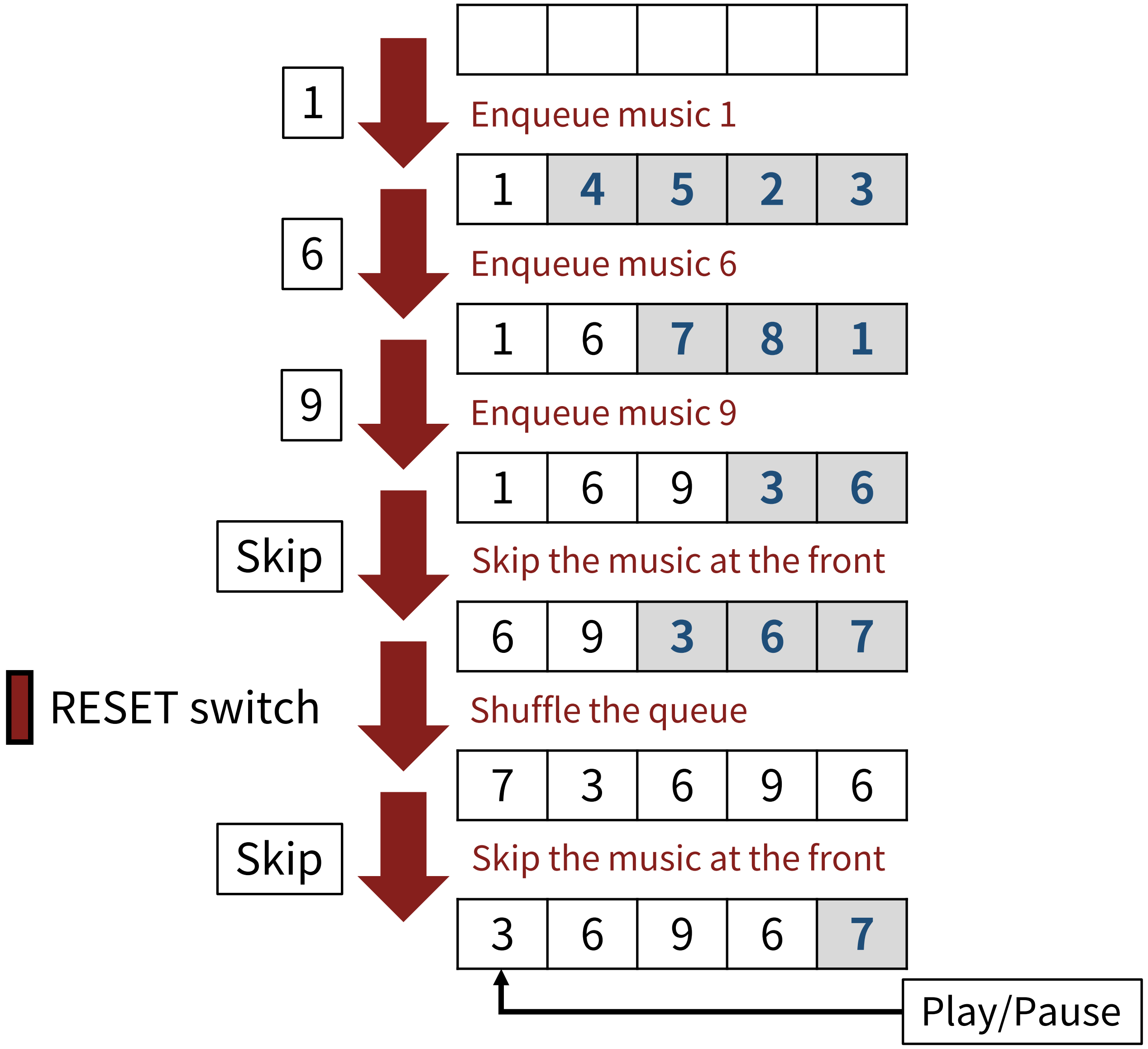
20211584 Junyeong Jang
Dept. of CS&E, Sogang Univ.
Embedded System Software – Final Project

Program Design

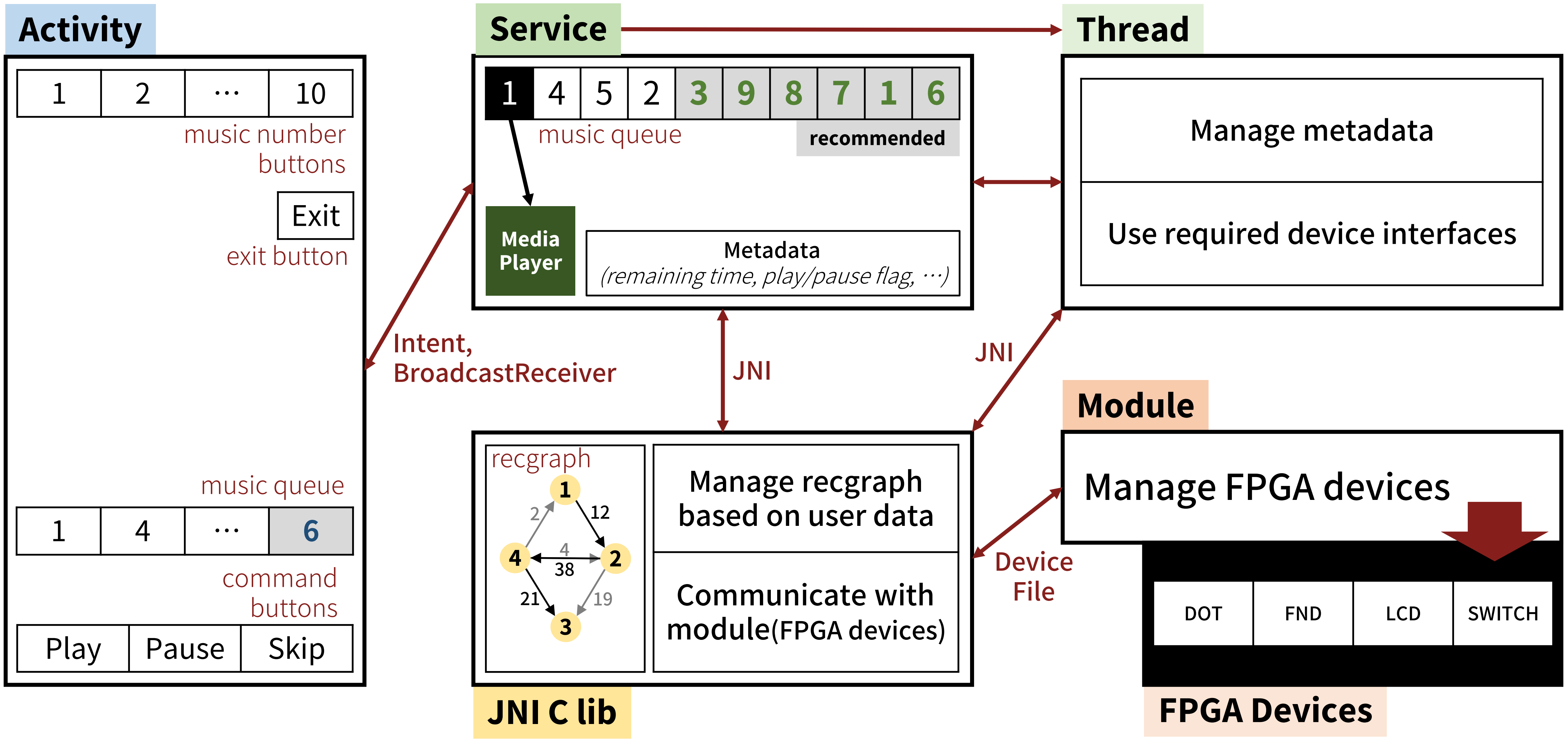




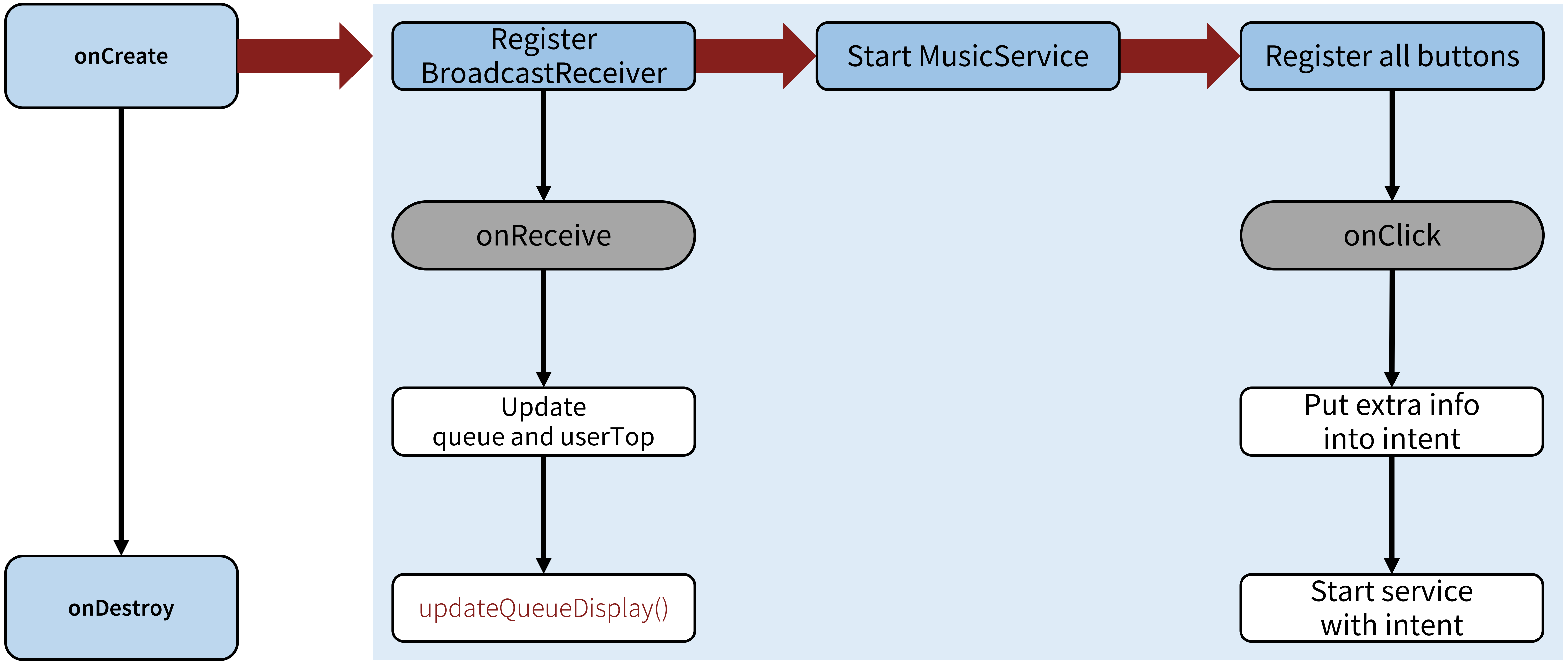
Program Design



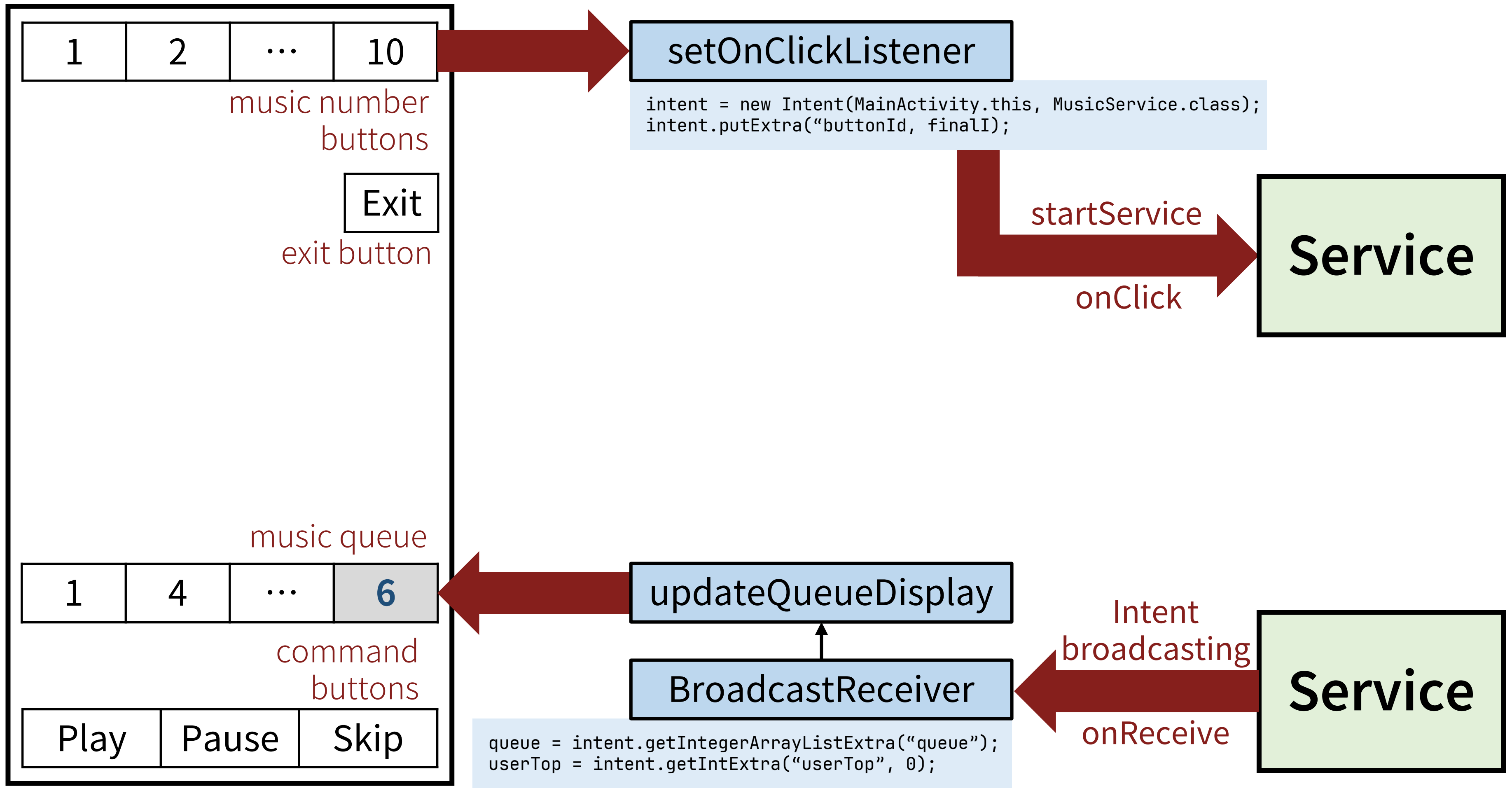
Program Design



Details - Activity



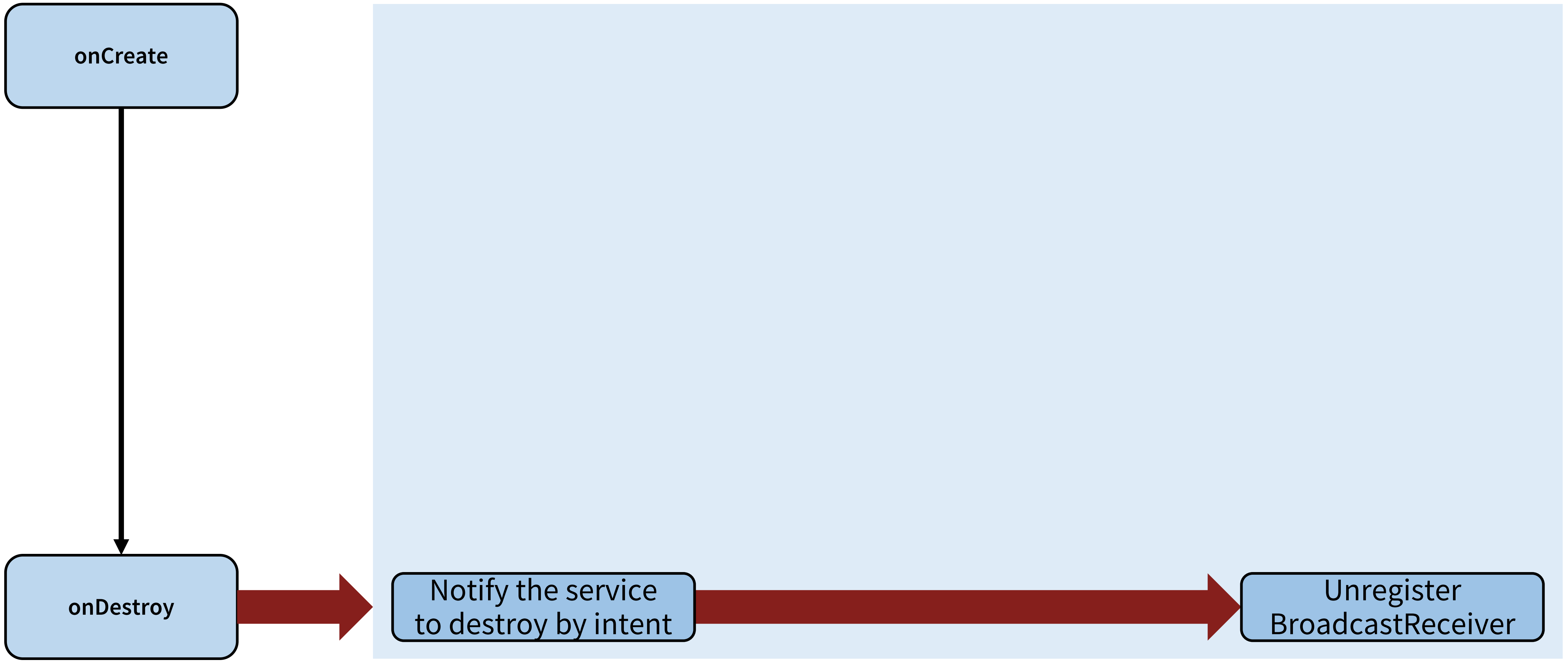
Details - Activity



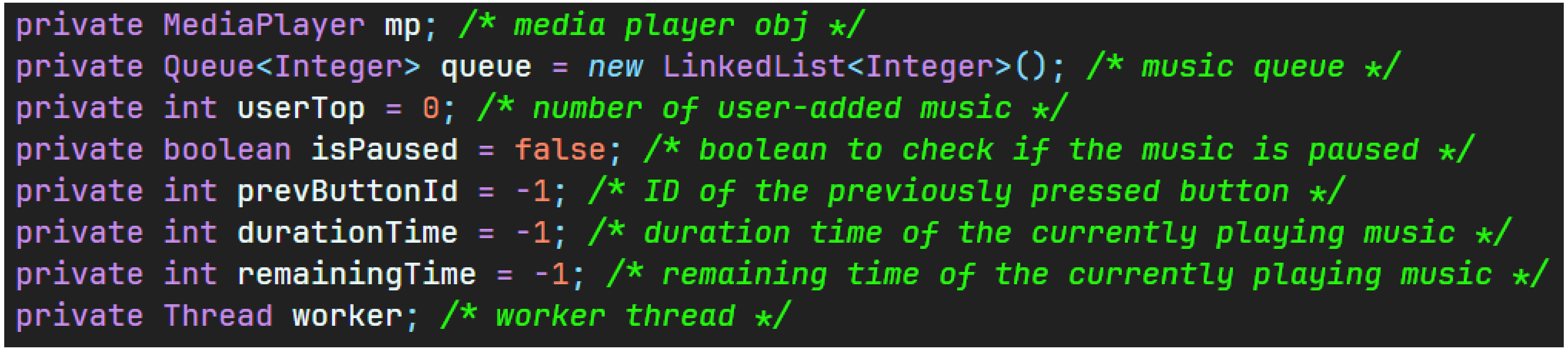
```
intent = new Intent(MainActivity.this, MusicService.class);
intent.putExtra("buttonId, finalI);
```

```
queue = intent.getIntegerArrayListExtra("queue");
userTop = intent.getIntExtra("userTop", 0);
```

Details - Activity


onCreate
onDestroy
Notify the service to destroy by intent
Unregister BroadcastReceiver

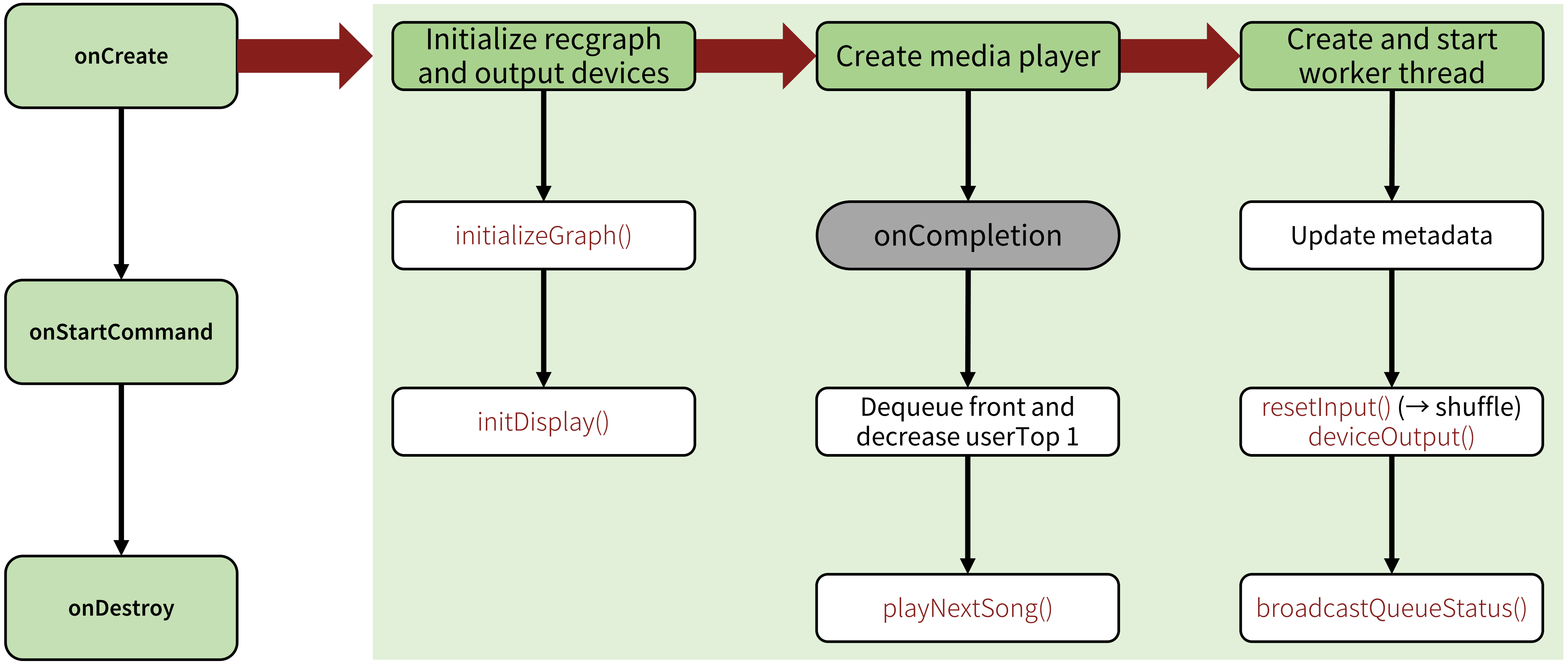
Details - Service



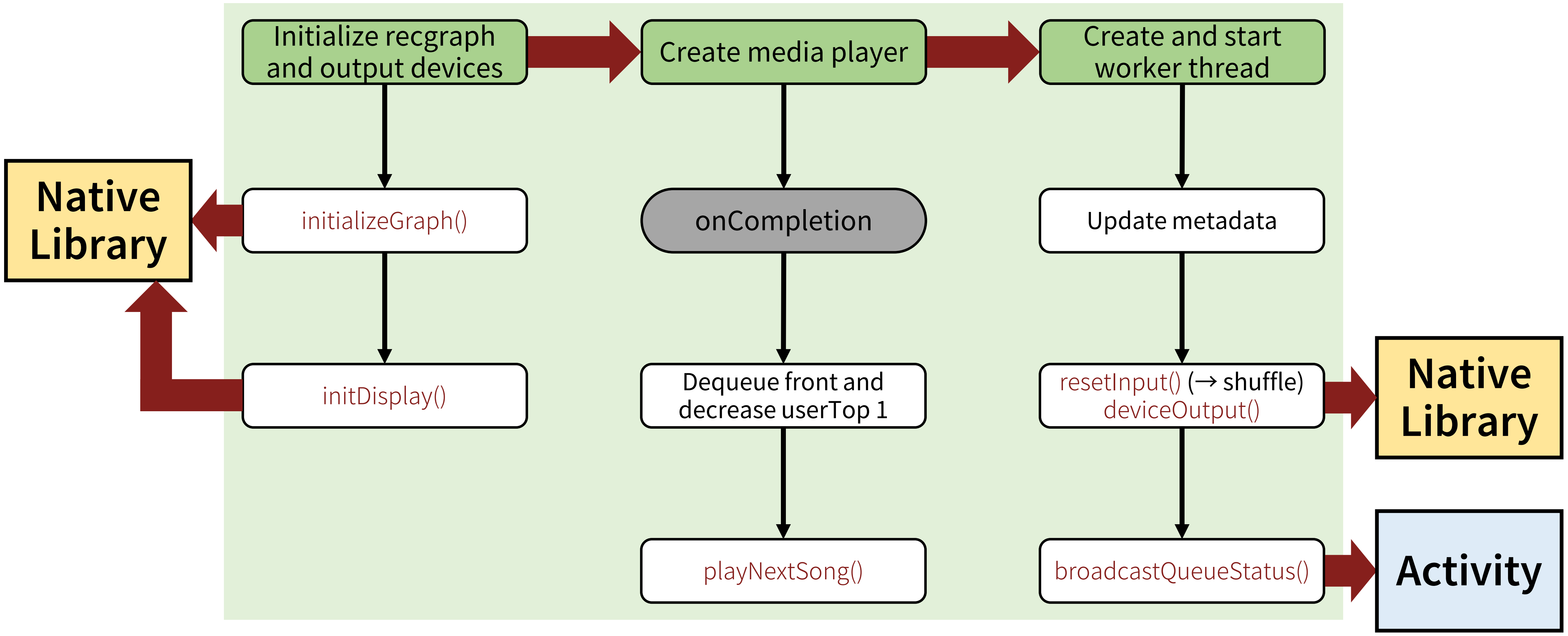
```
private MediaPlayer mp; /* media player obj */
private Queue<Integer> queue = new LinkedList<Integer>(); /* music queue */
private int userTop = 0; /* number of user-added music */
private boolean isPaused = false; /* boolean to check if the music is paused */
private int prevButtonId = -1; /* ID of the previously pressed button */
private int durationTime = -1; /* duration time of the currently playing music */
private int remainingTime = -1; /* remaining time of the currently playing music */
private Thread worker; /* worker thread */
```

MusicService

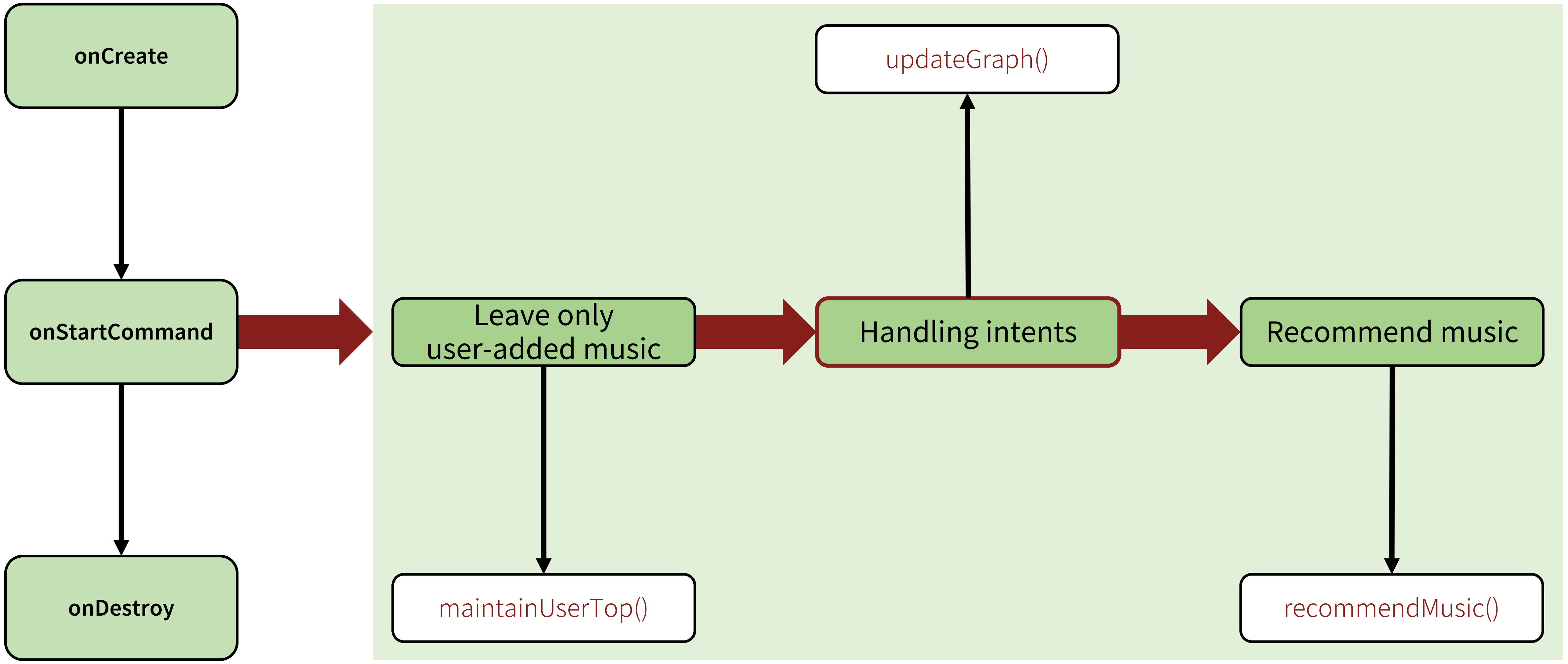
Details - Service



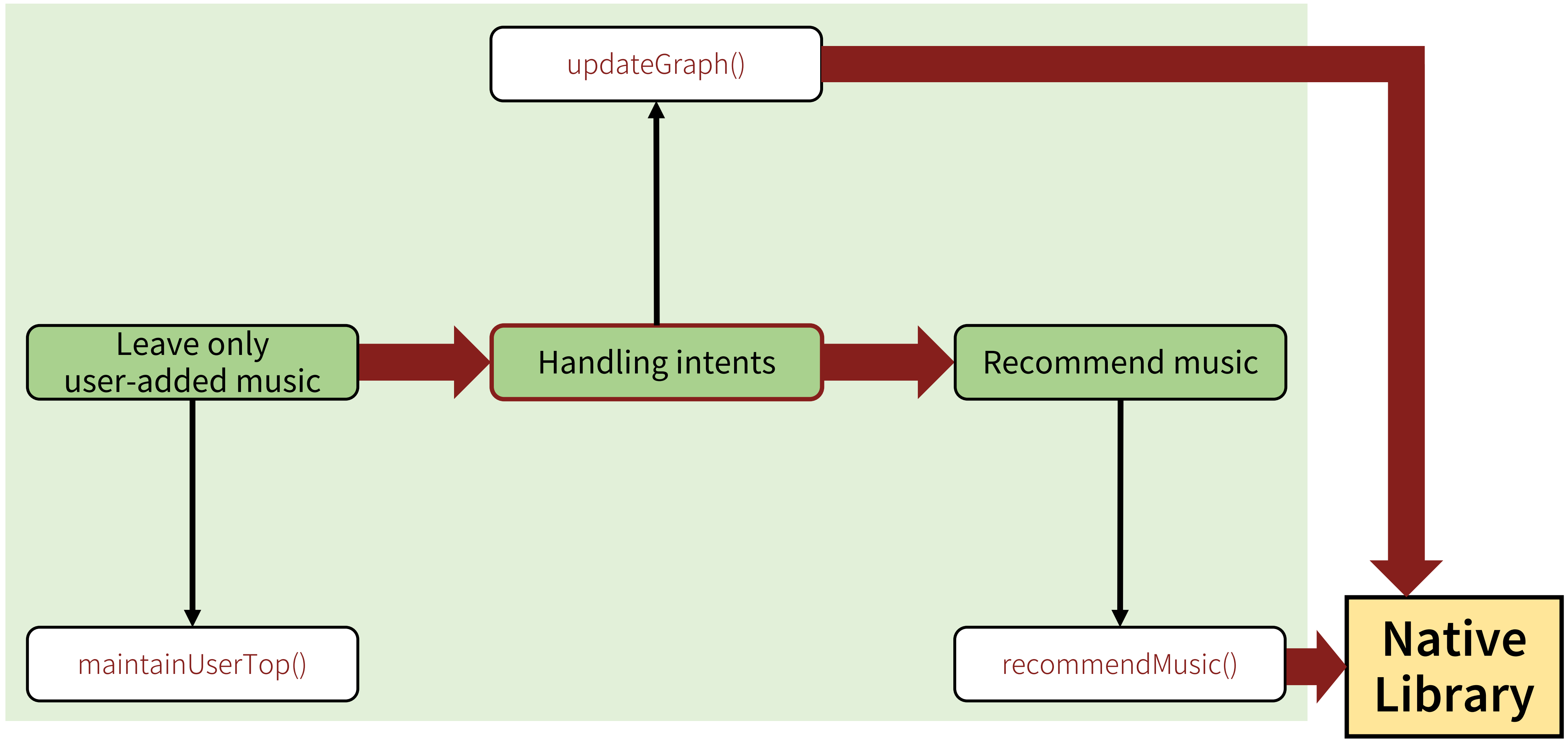
Details - Service



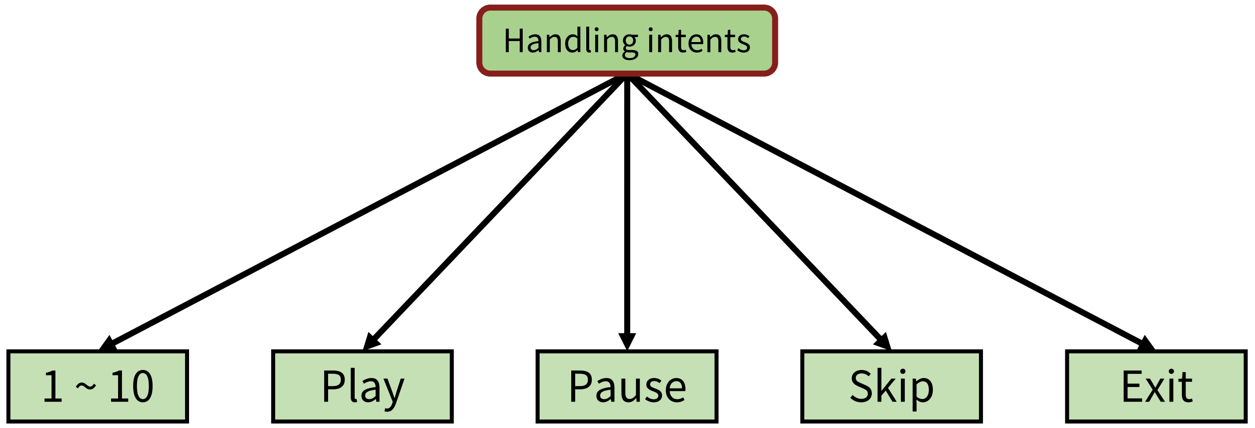
Details - Service

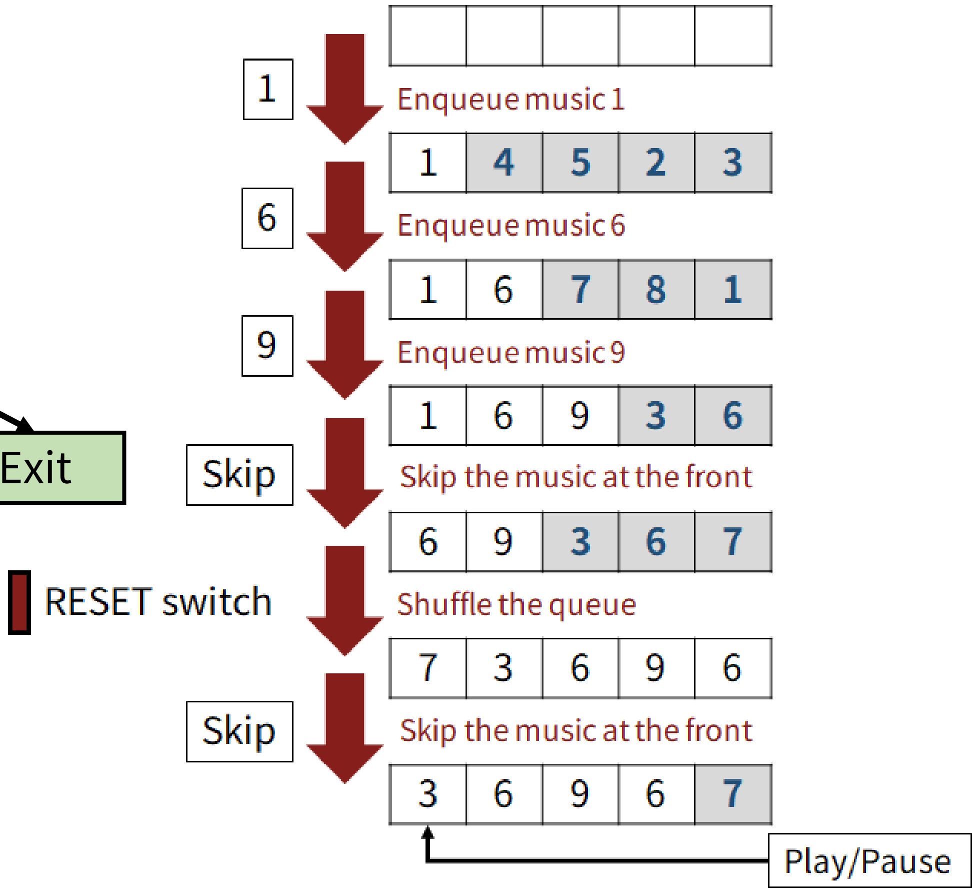


Details - Service

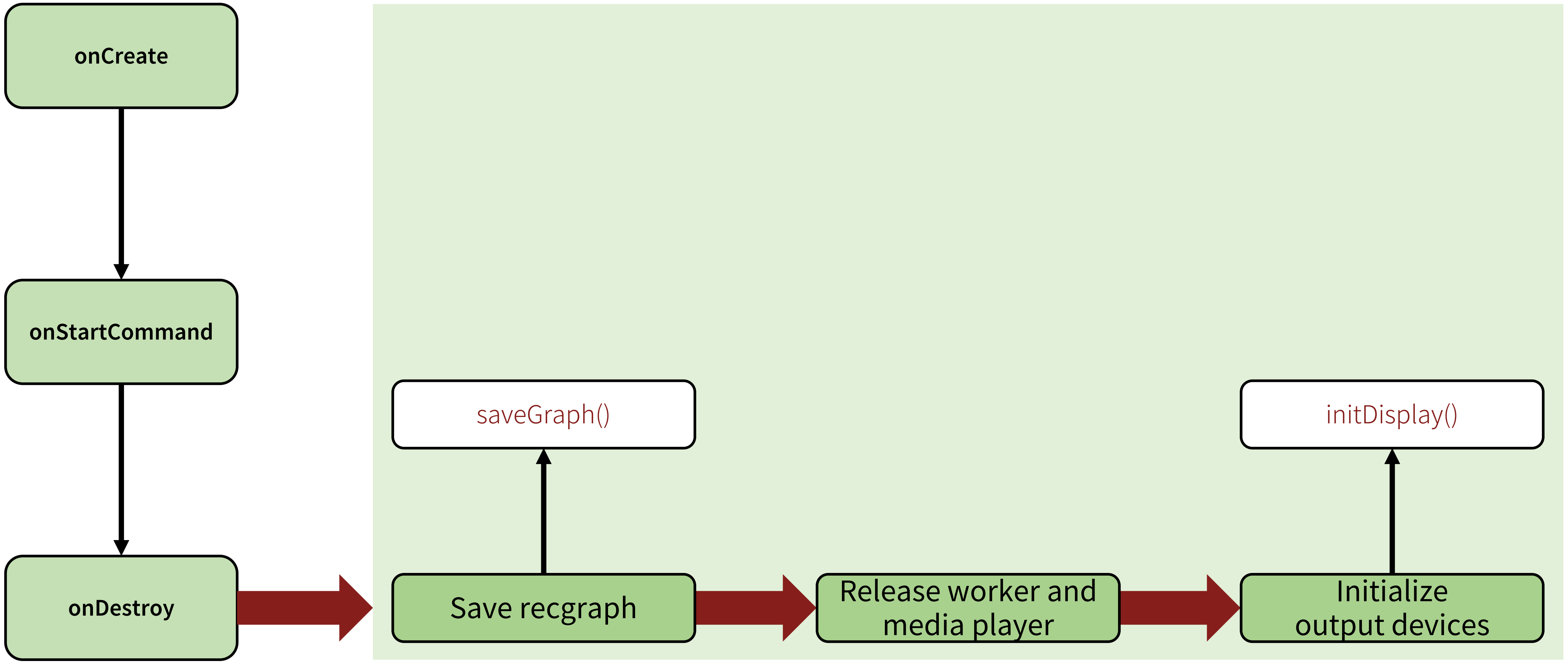


Details - Service

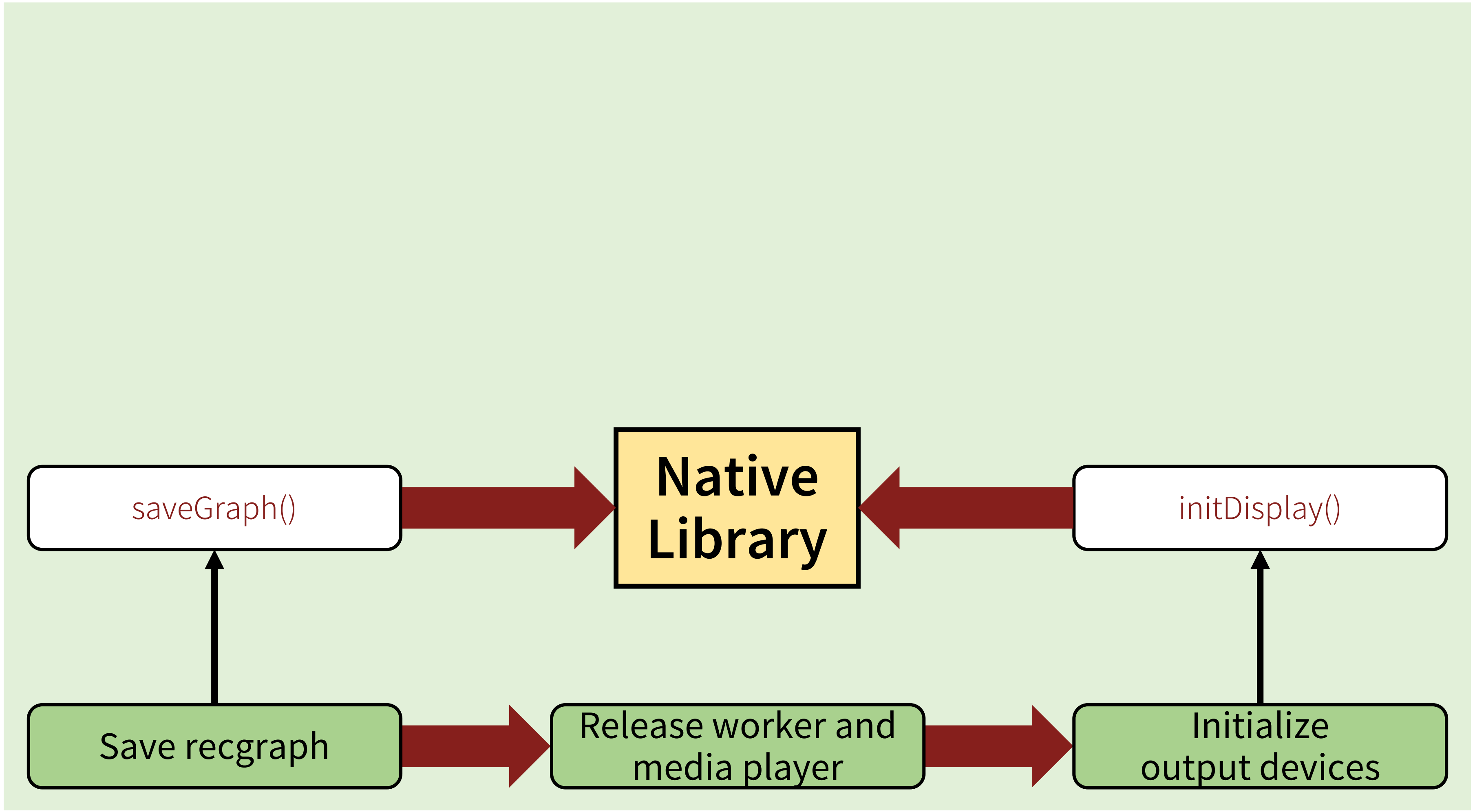




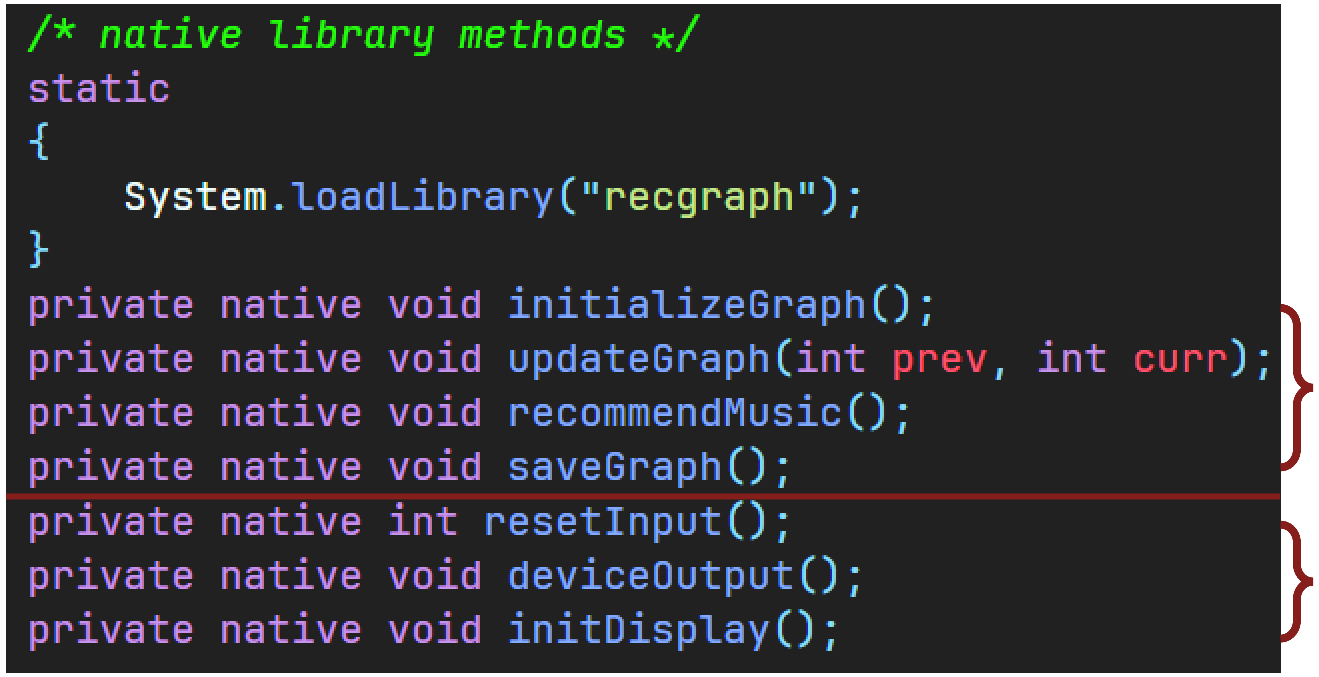
Details - Service



Details - Service



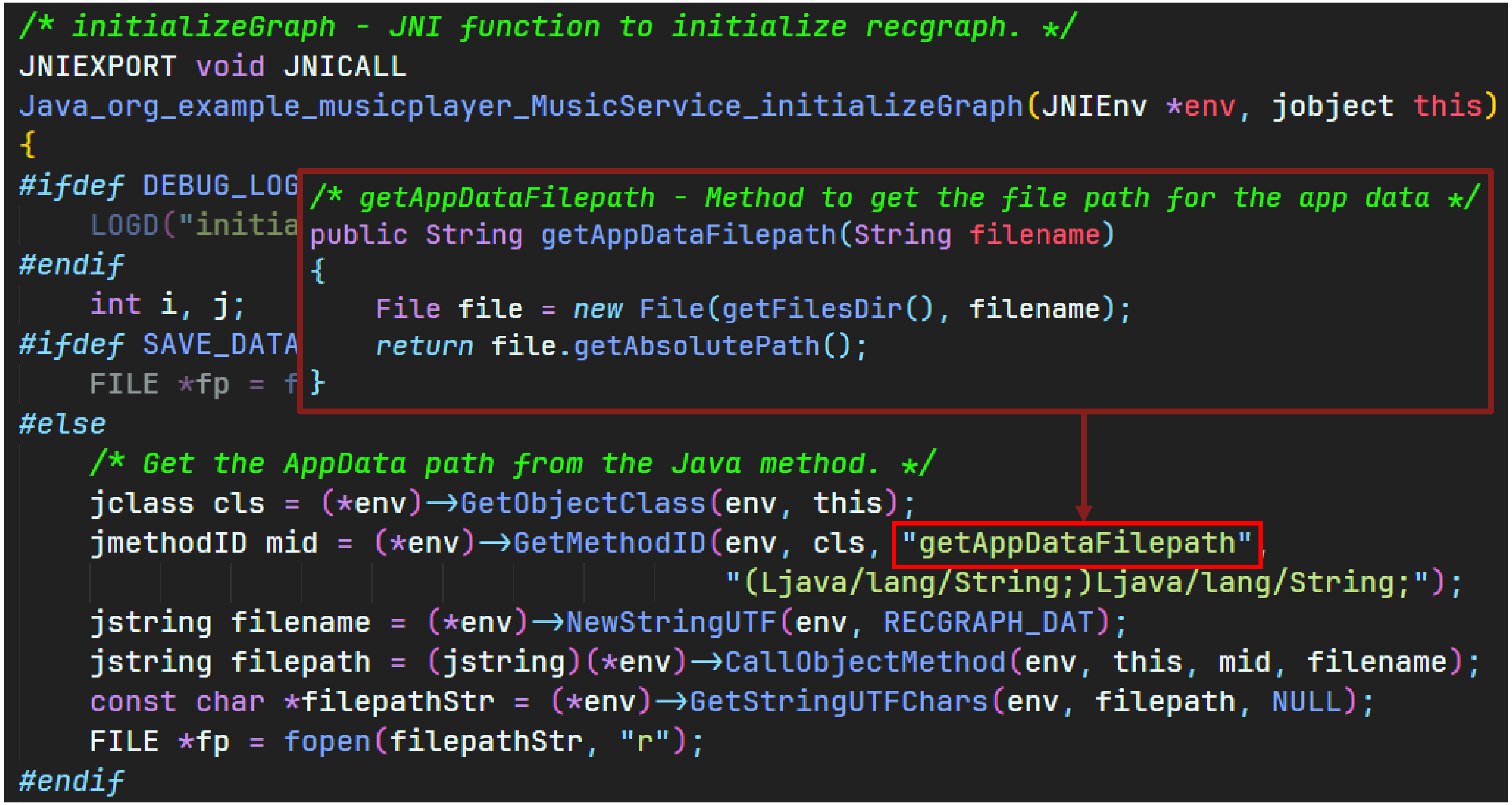
Details - C/C++ Native Library



```
/* native library methods */
static
{
    System.loadLibrary("recgraph");
}
private native void initializeGraph();
private native void updateGraph(int prev, int curr);
private native void recommendMusic();
private native void saveGraph();
private native int resetInput();
private native void deviceOutput();
private native void initDisplay();
```

MusicService

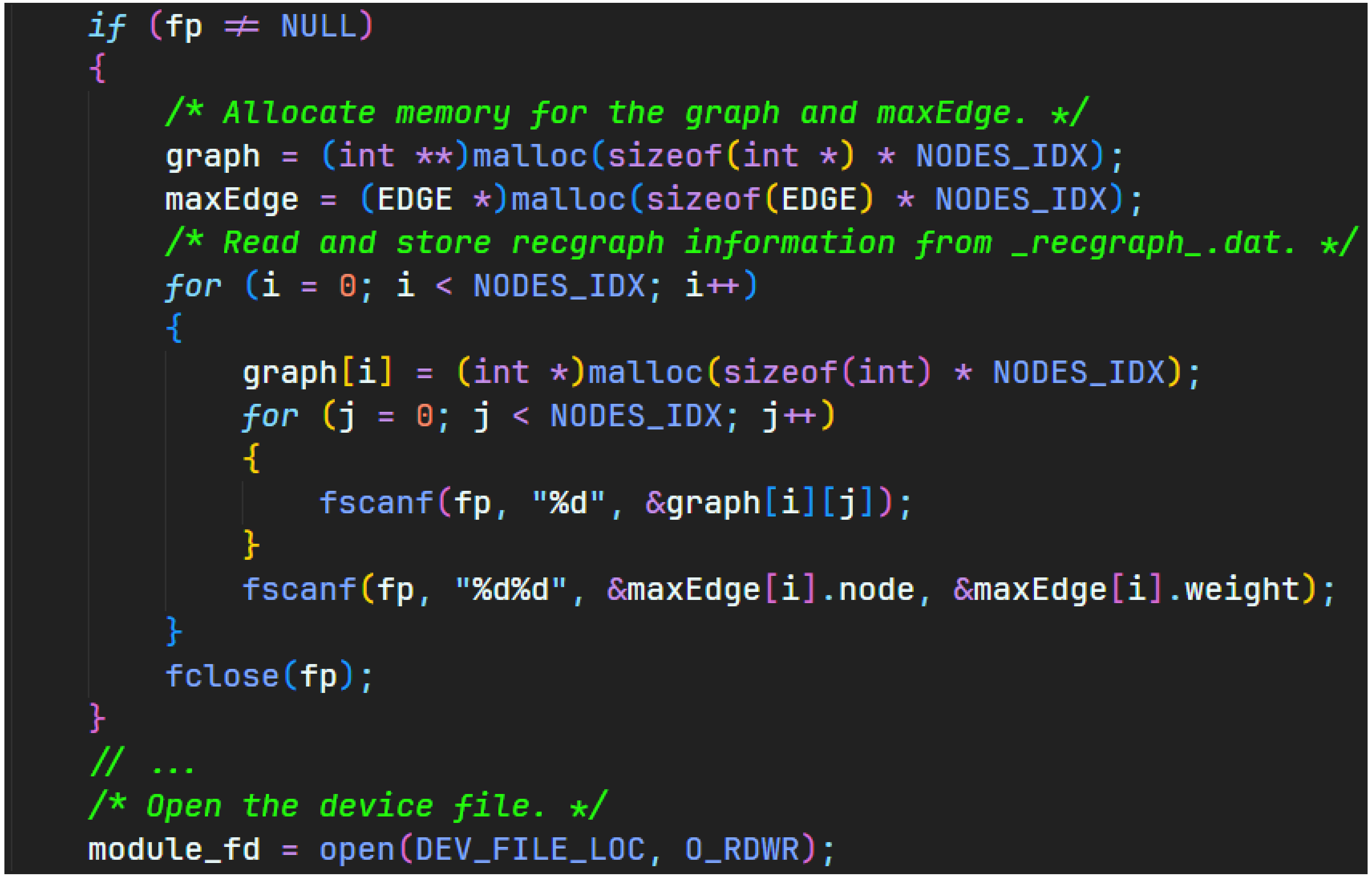
Details - C/C++ Native Library - initializeGraph



```c
/* initializeGraph - JNI function to initialize recgraph. */
JNIEXPORT void JNICALL
Java_org_example_musicplayer_MusicService_initializeGraph(JNIEnv *env, jobject this)
{
#ifdef DEBUG_LOG
    LOGD("initia
#endif
    int i, j;
#ifdef SAVE_DATA
    FILE *fp = f
#else
    /* Get the AppData path from the Java method. */
    jclass cls = (*env)->GetObjectClass(env, this);
    jmethodID mid = (*env)->GetMethodID(env, cls, "getAppDataFilepath",
                                        "(Ljava/lang/String;)Ljava/lang/String;");
    jstring filename = (*env)->NewStringUTF(env, RECGRAPH_DAT);
    jstring filepath = (jstring)(*env)->CallObjectMethod(env, this, mid, filename);
    const char *filepathStr = (*env)->GetStringUTFChars(env, filepath, NULL);
    FILE *fp = fopen(filepathStr, "r");
#endif
```

```java
/* getAppDataFilepath - Method to get the file path for the app data */
public String getAppDataFilepath(String filename)
{
    File file = new File(getFilesDir(), filename);
    return file.getAbsolutePath();
}
```

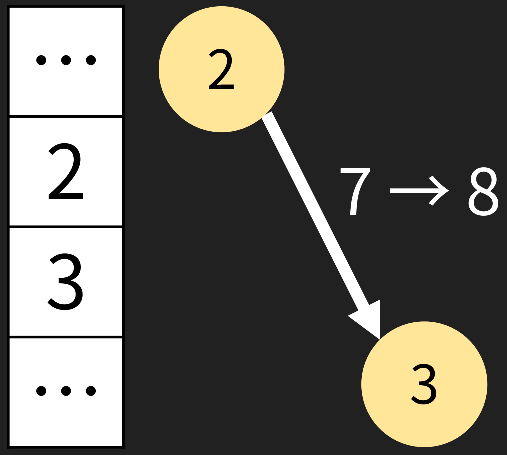
Details - C/C++ Native Library - initializeGraph



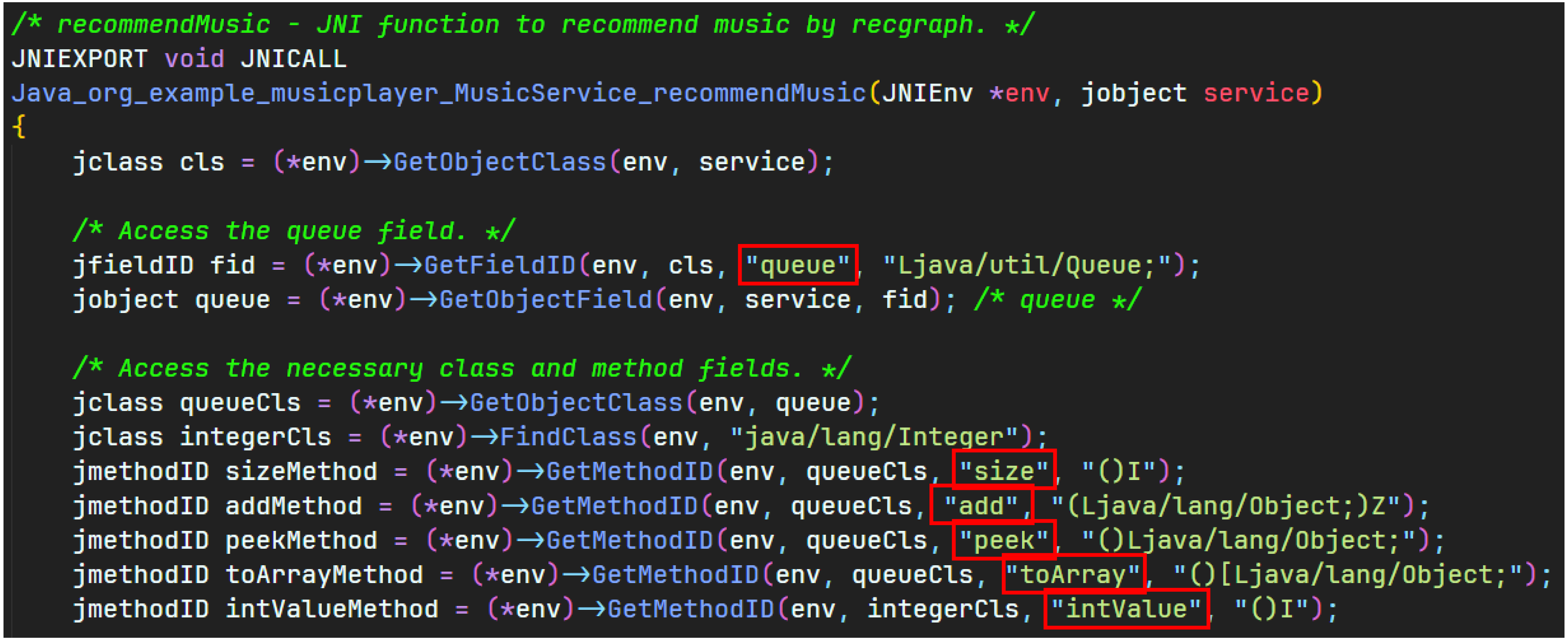
```c
if (fp != NULL)
{
    /* Allocate memory for the graph and maxEdge. */
    graph = (int **)malloc(sizeof(int *) * NODES_IDX);
    maxEdge = (EDGE *)malloc(sizeof(EDGE) * NODES_IDX);
    /* Read and store recgraph information from _recgraph_.dat. */
    for (i = 0; i < NODES_IDX; i++)
    {
        graph[i] = (int *)malloc(sizeof(int) * NODES_IDX);
        for (j = 0; j < NODES_IDX; j++)
        {
            fscanf(fp, "%d", &graph[i][j]);
        }
        fscanf(fp, "%d%d", &maxEdge[i].node, &maxEdge[i].weight);
    }
    fclose(fp);
}
// ...
/* Open the device file. */
module_fd = open(DEV_FILE_LOC, O_RDWR);
```

Details - C/C++ Native Library - updateGraph

```c
/* updateGraph - JNI function to update weights. */
JNIEXPORT void JNICALL
Java_org_example_musicplayer_MusicService_updateGraph(JNIEnv *env, jobject this,
                                                      jint prev, jint curr)
{
    /* Increases the edge weight by 1 between consecutively added music. */
    if (graph[prev][curr] < INT_MAX)
        graph[prev][curr]++;
    /* Update the maxEdge. */
    if (graph[prev][curr] > maxEdge[prev].weight)
    {
        maxEdge[prev].node = curr;
        maxEdge[prev].weight = graph[prev][curr];
    }
#ifdef DEBUG_LOGGING
    printGraph();
#endif
}
```

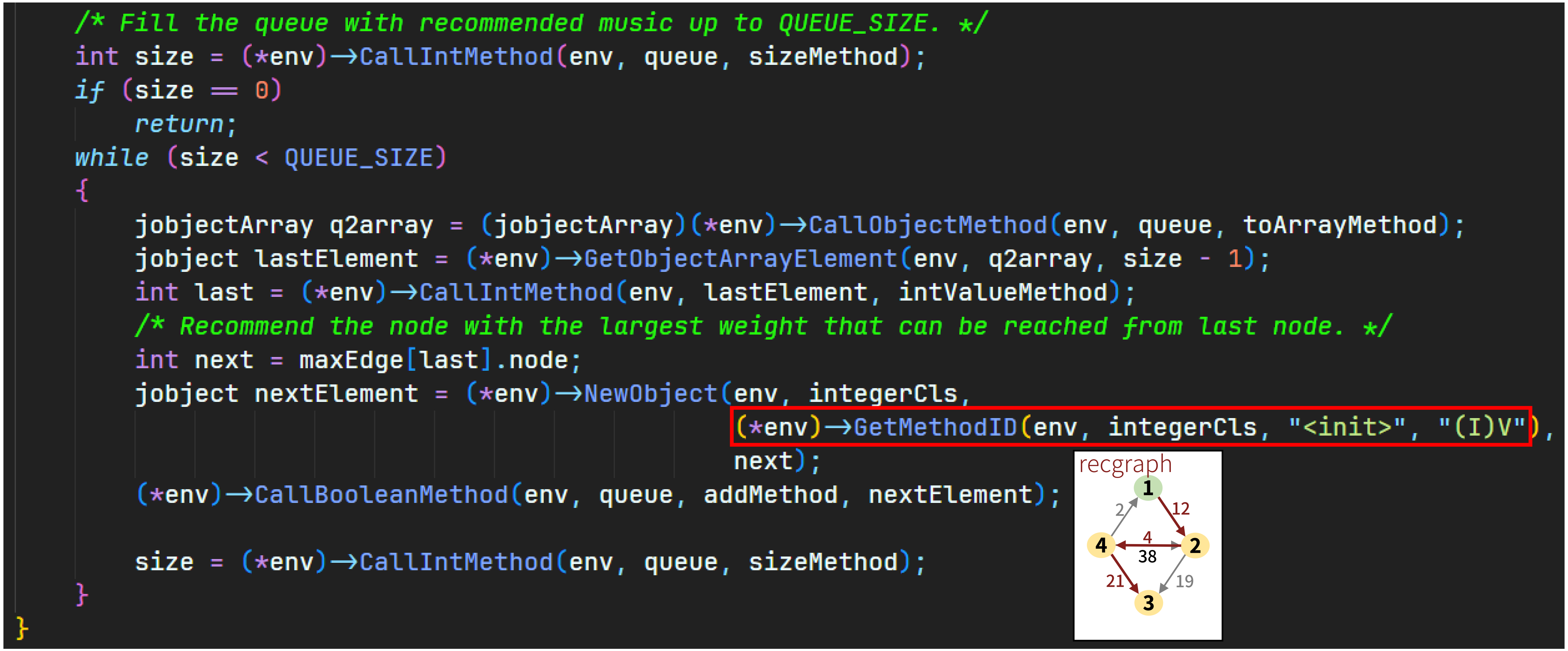


Details - C/C++ Native Library - recommendMusic

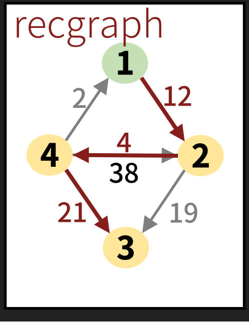


```
/* recommendMusic - JNI function to recommend music by recgraph. */
JNIEXPORT void JNICALL
Java_org_example_musicplayer_MusicService_recommendMusic(JNIEnv *env, jobject service)
{
    jclass cls = (*env)->GetObjectClass(env, service);

    /* Access the queue field. */
    jfieldID fid = (*env)->GetFieldID(env, cls, "queue", "Ljava/util/Queue;");
    jobject queue = (*env)->GetObjectField(env, service, fid); /* queue */

    /* Access the necessary class and method fields. */
    jclass queueCls = (*env)->GetObjectClass(env, queue);
    jclass integerCls = (*env)->FindClass(env, "java/lang/Integer");
    jmethodID sizeMethod = (*env)->GetMethodID(env, queueCls, "size", "()I");
    jmethodID addMethod = (*env)->GetMethodID(env, queueCls, "add", "(Ljava/lang/Object;)Z");
    jmethodID peekMethod = (*env)->GetMethodID(env, queueCls, "peek", "()Ljava/lang/Object;");
    jmethodID toArrayMethod = (*env)->GetMethodID(env, queueCls, "toArray", "()[Ljava/lang/Object;");
    jmethodID intValueMethod = (*env)->GetMethodID(env, integerCls, "intValue", "()I");
```

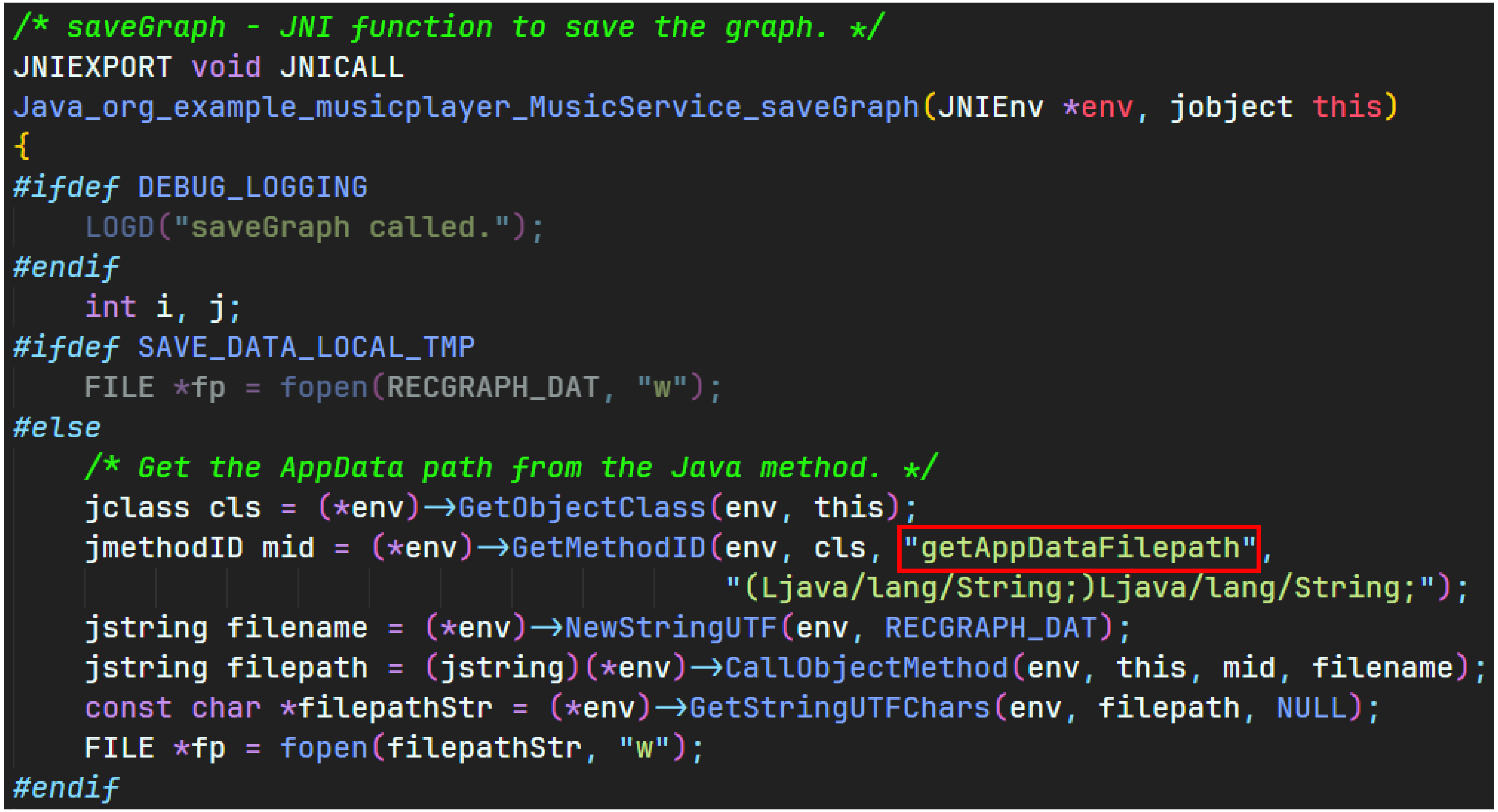
Details - C/C++ Native Library - recommendMusic



```
    /* Fill the queue with recommended music up to QUEUE_SIZE. */
    int size = (*env)->CallIntMethod(env, queue, sizeMethod);
    if (size == 0)
        return;
    while (size < QUEUE_SIZE)
    {
        jobjectArray q2array = (jobjectArray)(*env)->CallObjectMethod(env, queue, toArrayMethod);
        jobject lastElement = (*env)->GetObjectArrayElement(env, q2array, size - 1);
        int last = (*env)->CallIntMethod(env, lastElement, intValueMethod);
        /* Recommend the node with the largest weight that can be reached from last node. */
        int next = maxEdge[last].node;
        jobject nextElement = (*env)->NewObject(env, integerCls,
                                                (*env)->GetMethodID(env, integerCls, "<init>", "(I)V"),
                                                next);
        (*env)->CallBooleanMethod(env, queue, addMethod, nextElement);

        size = (*env)->CallIntMethod(env, queue, sizeMethod);
    }
}
```

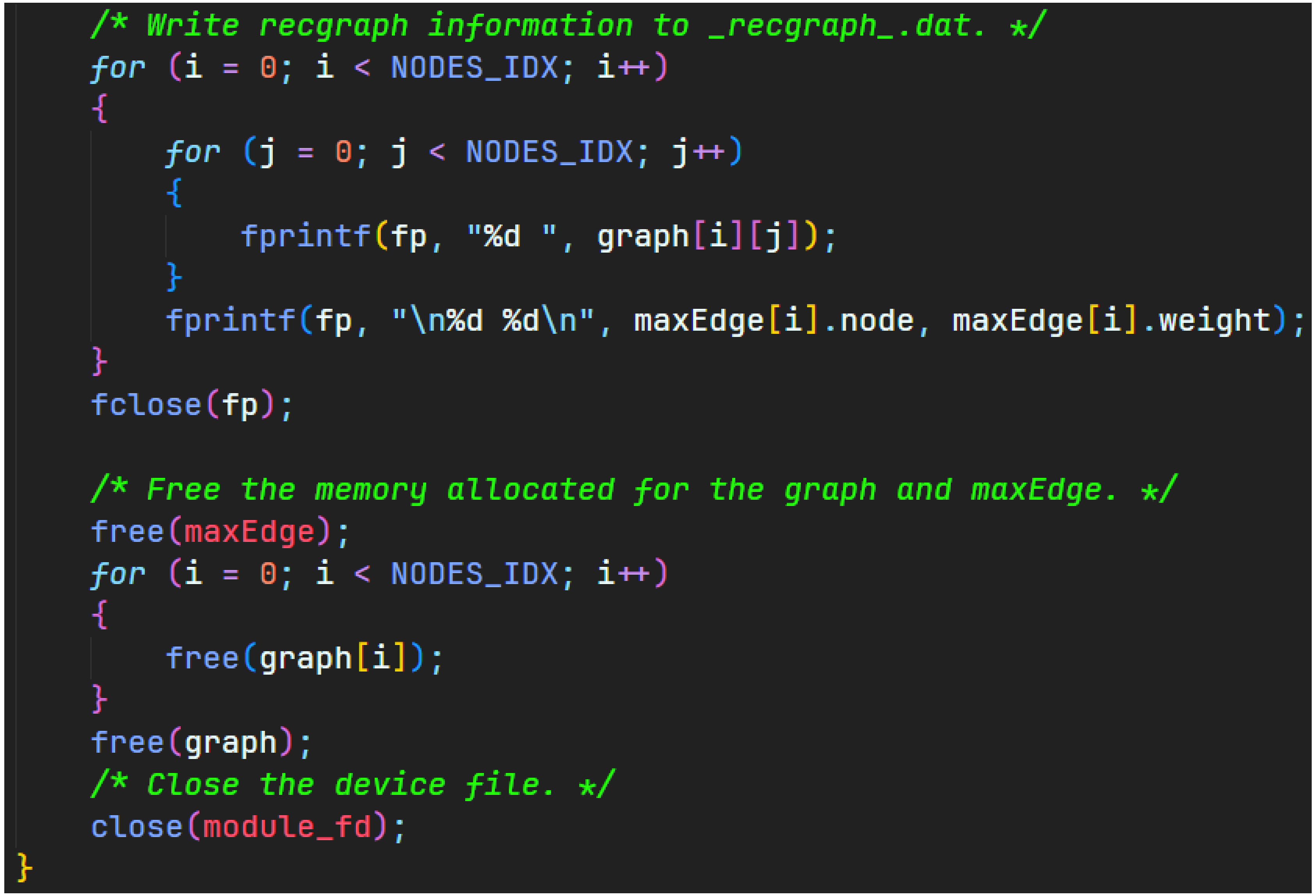


Details - C/C++ Native Library - saveGraph



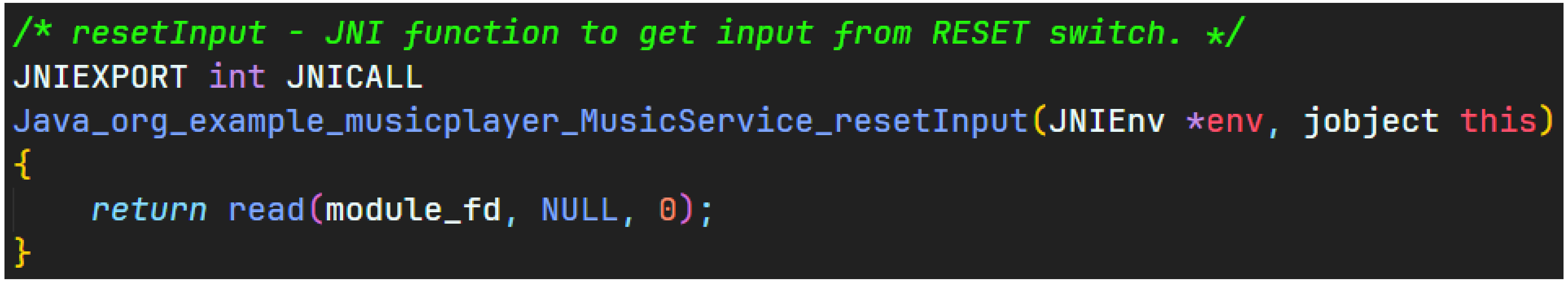
```
/* saveGraph - JNI function to save the graph. */
JNIEXPORT void JNICALL
Java_org_example_musicplayer_MusicService_saveGraph(JNIEnv *env, jobject this)
{
#ifdef DEBUG_LOGGING
    LOGD("saveGraph called.");
#endif
    int i, j;
#ifdef SAVE_DATA_LOCAL_TMP
    FILE *fp = fopen(RECGRAPH_DAT, "w");
#else
    /* Get the AppData path from the Java method. */
    jclass cls = (*env)->GetObjectClass(env, this);
    jmethodID mid = (*env)->GetMethodID(env, cls, "getAppDataFilepath",
                                        "(Ljava/lang/String;)Ljava/lang/String;");
    jstring filename = (*env)->NewStringUTF(env, RECGRAPH_DAT);
    jstring filepath = (jstring)(*env)->CallObjectMethod(env, this, mid, filename);
    const char *filepathStr = (*env)->GetStringUTFChars(env, filepath, NULL);
    FILE *fp = fopen(filepathStr, "w");
#endif
```

Details - C/C++ Native Library - saveGraph



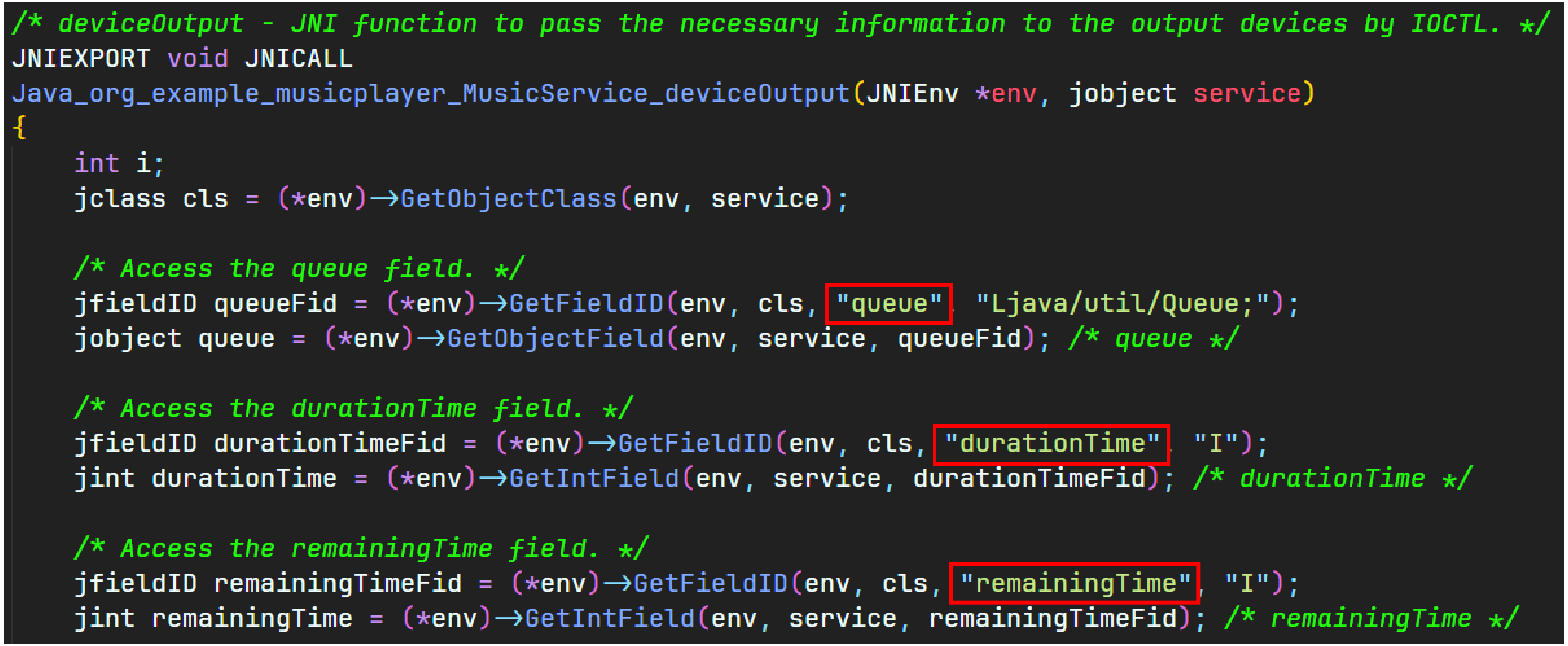
```
    /* Write recgraph information to _recgraph_.dat. */
    for (i = 0; i < NODES_IDX; i++)
    {
        for (j = 0; j < NODES_IDX; j++)
        {
            fprintf(fp, "%d ", graph[i][j]);
        }
        fprintf(fp, "\n%d %d\n", maxEdge[i].node, maxEdge[i].weight);
    }
    fclose(fp);

    /* Free the memory allocated for the graph and maxEdge. */
    free(maxEdge);
    for (i = 0; i < NODES_IDX; i++)
    {
        free(graph[i]);
    }
    free(graph);
    /* Close the device file. */
    close(module_fd);
}
```

Details - C/C++ Native Library - resetInput



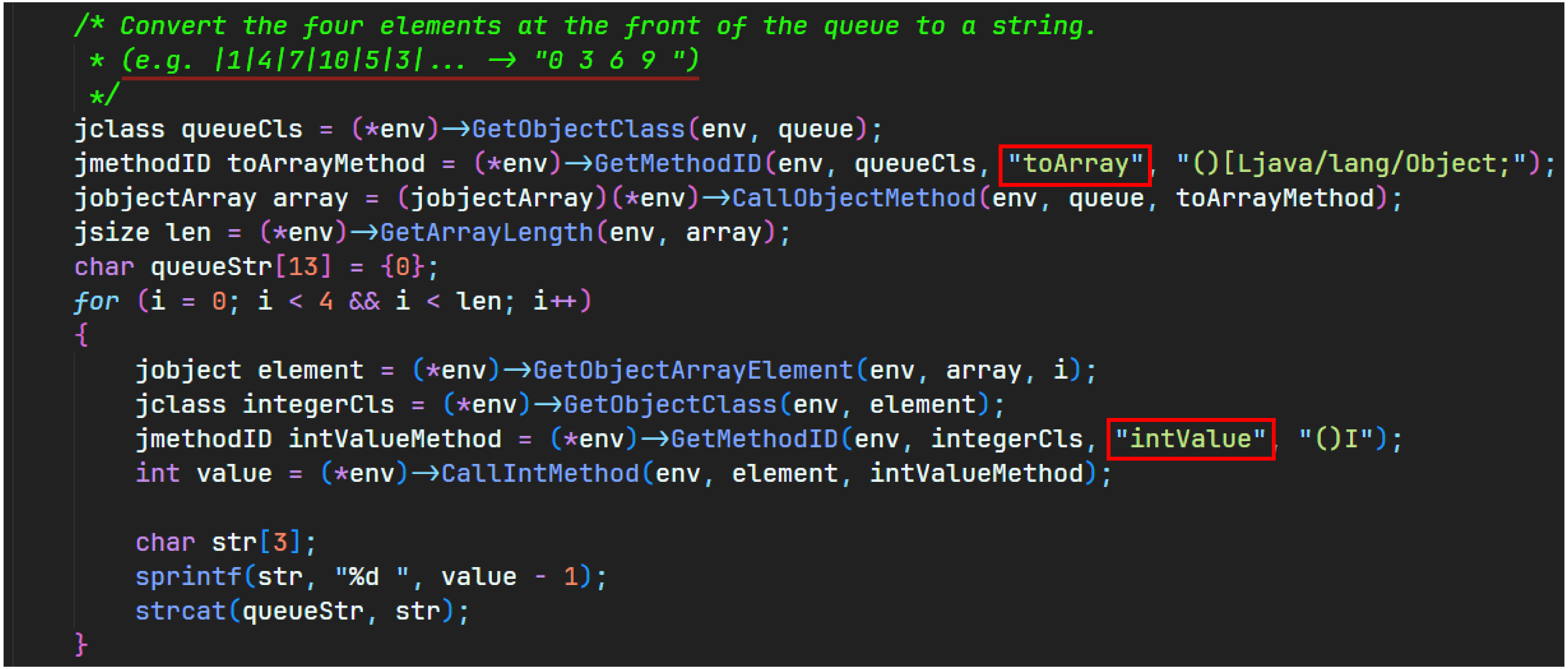
```
/* resetInput - JNI function to get input from RESET switch. */
JNIEXPORT int JNICALL
Java_org_example_musicplayer_MusicService_resetInput(JNIEnv *env, jobject this)
{
    return read(module_fd, NULL, 0);
}
```

Details - C/C++ Native Library - deviceOutput



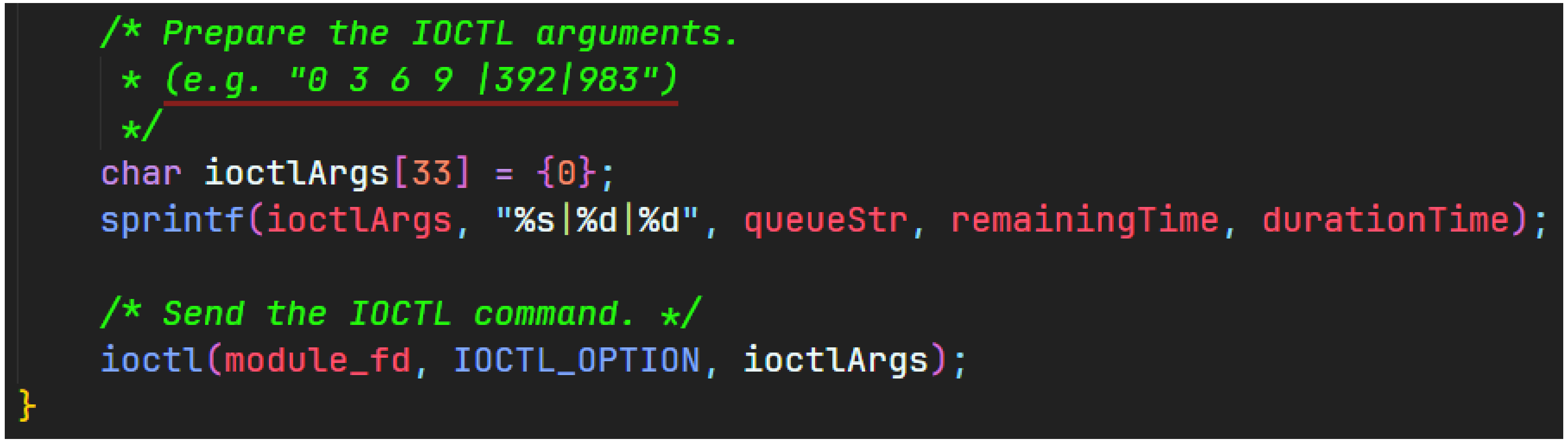
```
/* deviceOutput - JNI function to pass the necessary information to the output devices by IOCTL. */
JNIEXPORT void JNICALL
Java_org_example_musicplayer_MusicService_deviceOutput(JNIEnv *env, jobject service)
{
    int i;
    jclass cls = (*env)->GetObjectClass(env, service);

    /* Access the queue field. */
    jfieldID queueFid = (*env)->GetFieldID(env, cls, "queue", "Ljava/util/Queue;");
    jobject queue = (*env)->GetObjectField(env, service, queueFid); /* queue */

    /* Access the durationTime field. */
    jfieldID durationTimeFid = (*env)->GetFieldID(env, cls, "durationTime", "I");
    jint durationTime = (*env)->GetIntField(env, service, durationTimeFid); /* durationTime */

    /* Access the remainingTime field. */
    jfieldID remainingTimeFid = (*env)->GetFieldID(env, cls, "remainingTime", "I");
    jint remainingTime = (*env)->GetIntField(env, service, remainingTimeFid); /* remainingTime */
```

Details - C/C++ Native Library - deviceOutput



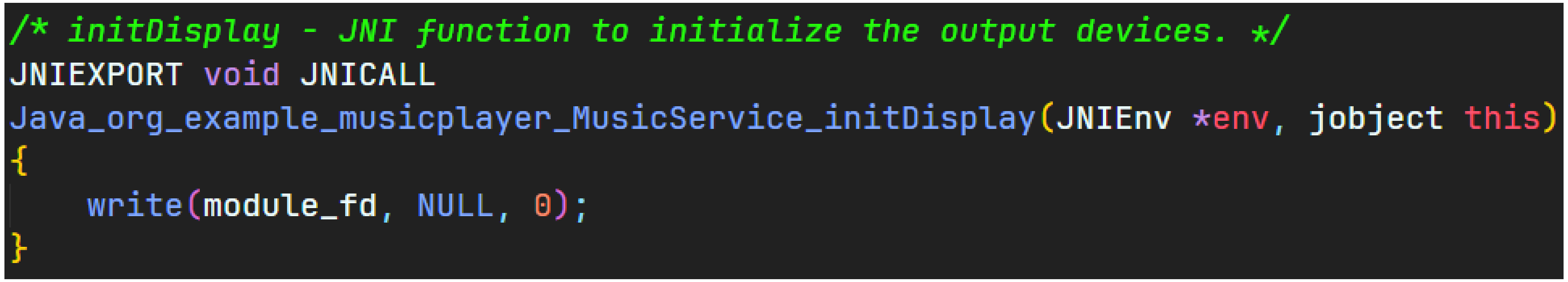
```
/* Convert the four elements at the front of the queue to a string.
 * (e.g. |1|4|7|10|5|3|...  → "0 3 6 9 ")
 */
jclass queueCls = (*env)→GetObjectClass(env, queue);
jmethodID toArrayMethod = (*env)→GetMethodID(env, queueCls, "toArray", "()[Ljava/lang/Object;");
jobjectArray array = (jobjectArray)(*env)→CallObjectMethod(env, queue, toArrayMethod);
jsize len = (*env)→GetArrayLength(env, array);
char queueStr[13] = {0};
for (i = 0; i < 4 && i < len; i++)
{
    jobject element = (*env)→GetObjectArrayElement(env, array, i);
    jclass integerCls = (*env)→GetObjectClass(env, element);
    jmethodID intValueMethod = (*env)→GetMethodID(env, integerCls, "intValue", "()I");
    int value = (*env)→CallIntMethod(env, element, intValueMethod);

    char str[3];
    sprintf(str, "%d ", value - 1);
    strcat(queueStr, str);
}
```

Details - C/C++ Native Library - deviceOutput



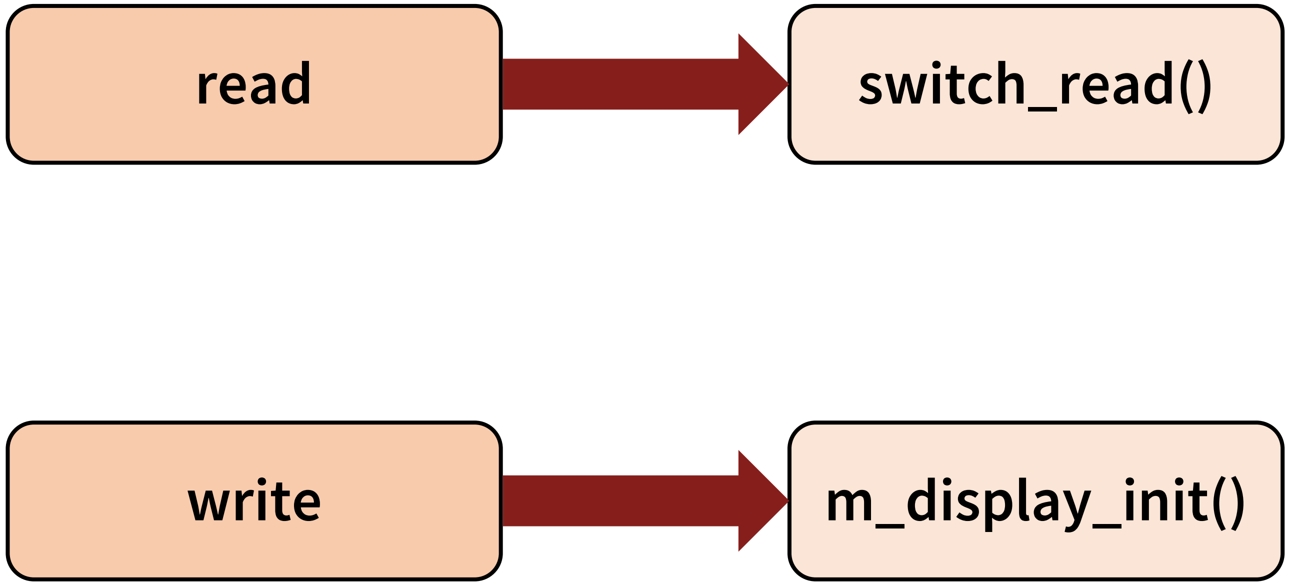
```
    /* Prepare the IOCTL arguments.
     * (e.g. "0 3 6 9 |392|983")
     */
    char ioctlArgs[33] = {0};
    sprintf(ioctlArgs, "%s|%d|%d", queueStr, remainingTime, durationTime);

    /* Send the IOCTL command. */
    ioctl(module_fd, IOCTL_OPTION, ioctlArgs);
}
```

Details - C/C++ Native Library - initDisplay

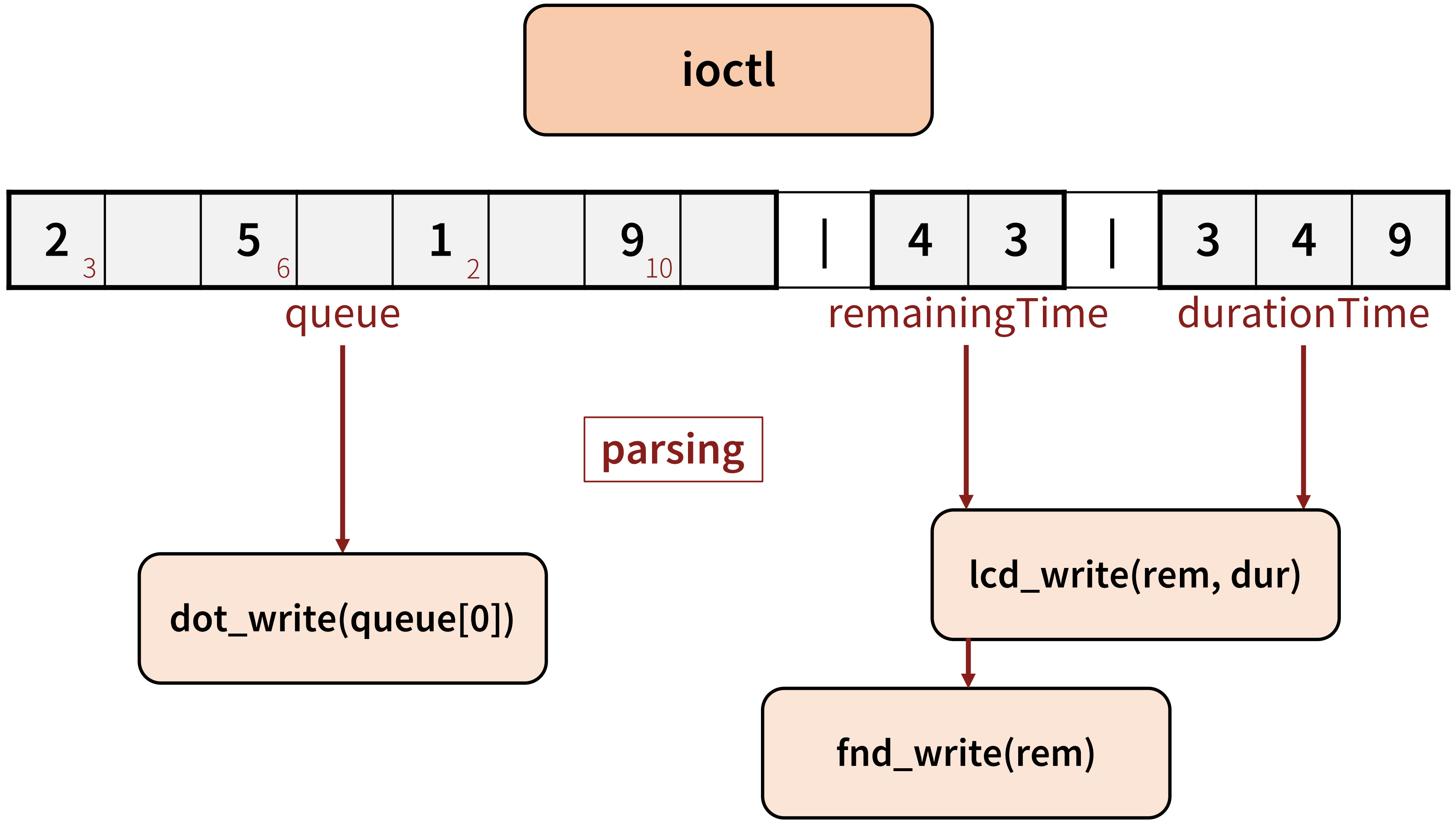


```
/* initDisplay - JNI function to initialize the output devices. */
JNIEXPORT void JNICALL
Java_org_example_musicplayer_MusicService_initDisplay(JNIEnv *env, jobject this)
{
    write(module_fd, NULL, 0);
}
```

Details - Linux Kernel Module



Details - Linux Kernel Module



Program Demonstration

```
$ cd /data/local/tmp
$ insmod music_driver.ko
$ mknod /dev/music_driver c 242 0
```

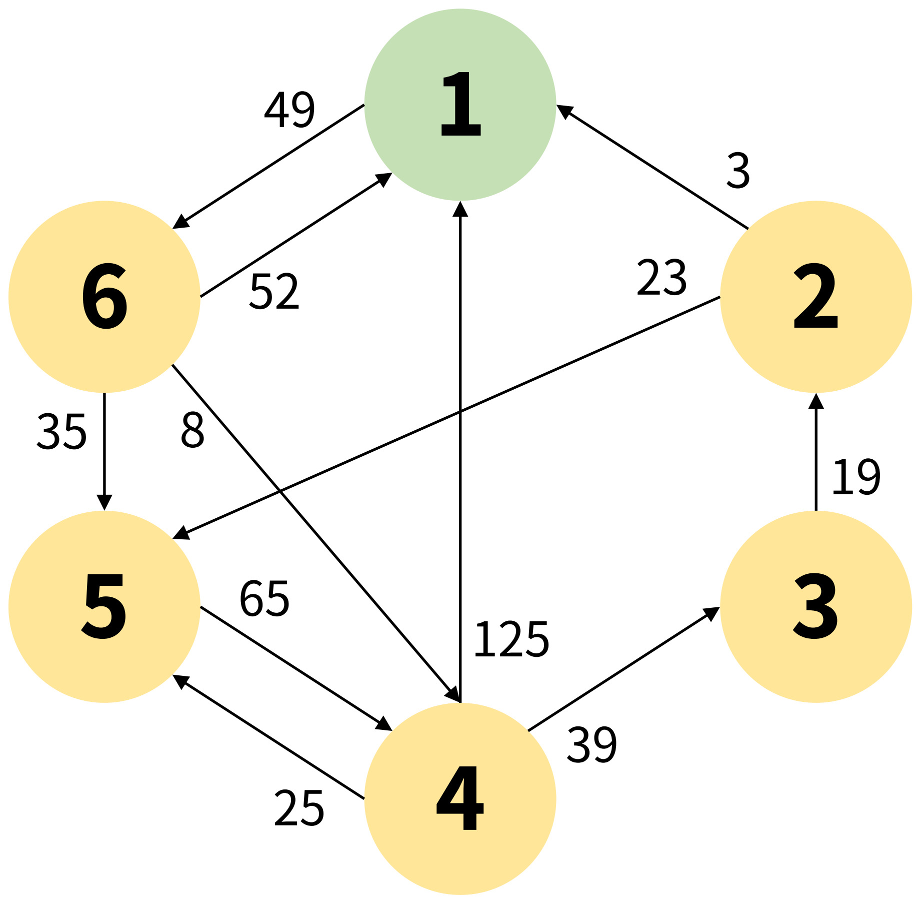
music1.mp3	4:47
music2.mp3	22:20
music3.mp3	3:55
music4.mp3	6:32
music5.mp3	2:19
music6.mp3	1:24
music7.mp3	5:51
music8.mp3	4:49
music9.mp3	4:56
music10.mp3	3:21

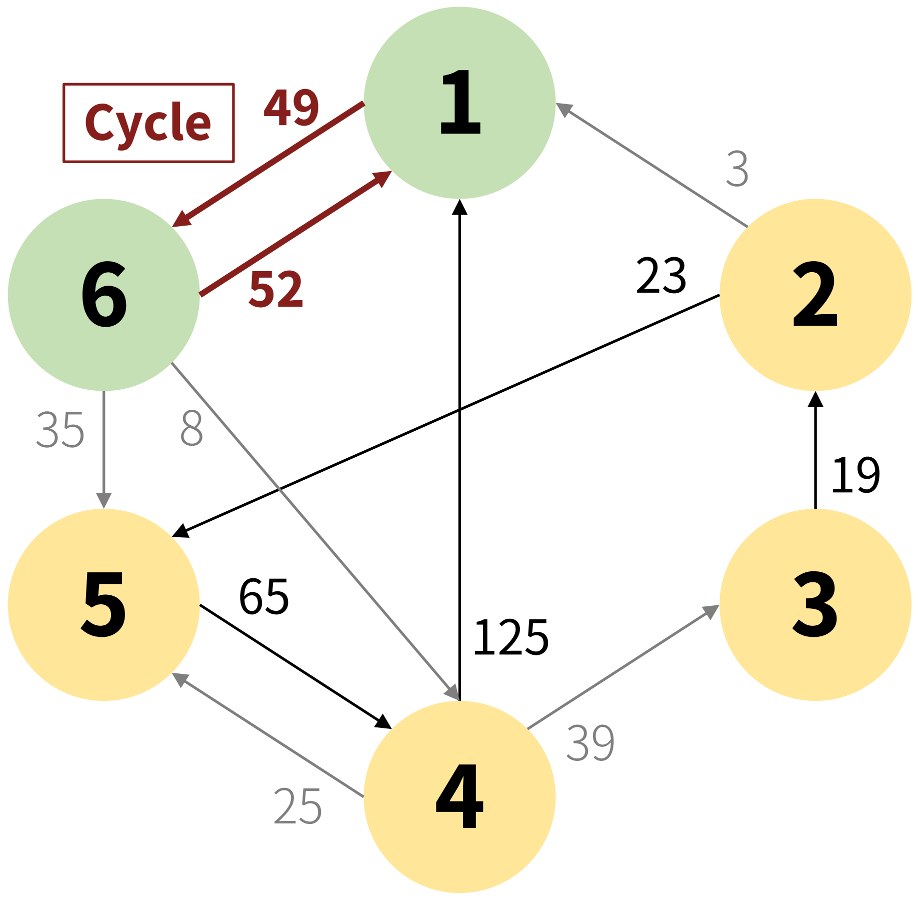
Follow-up Study Plan

What I focused on in this project is . . .

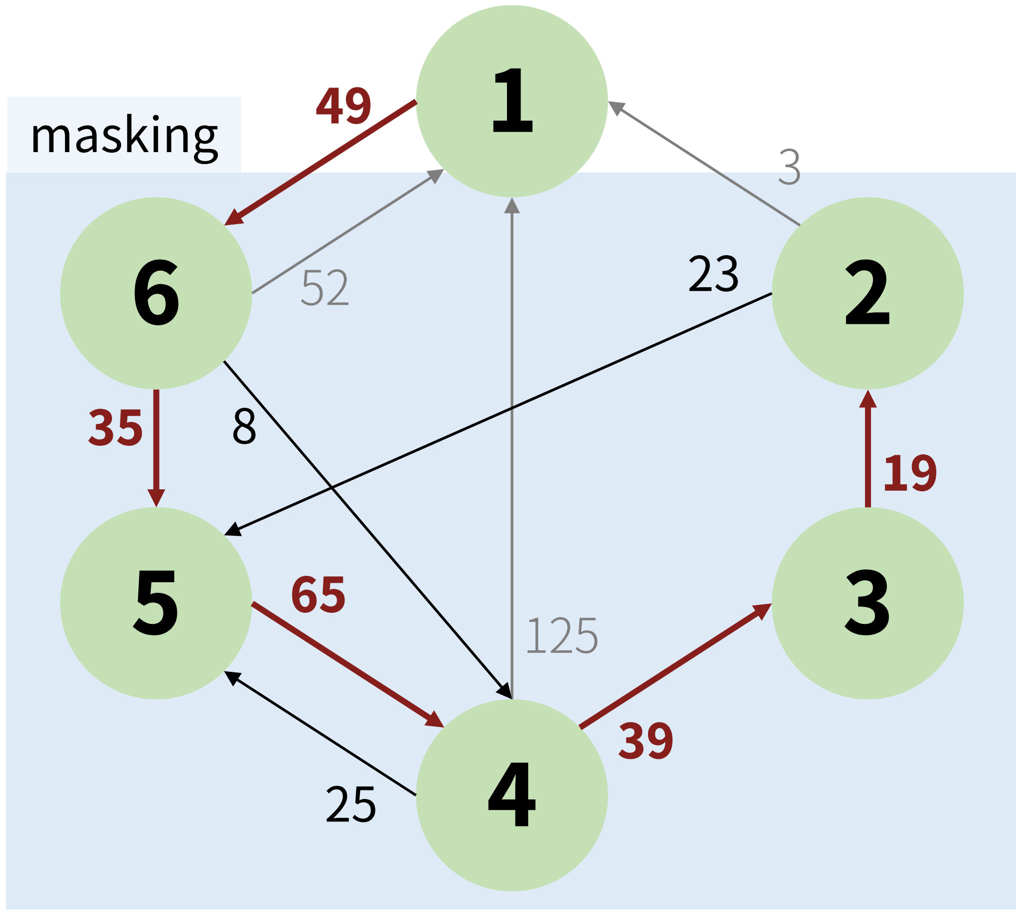
- Linux Kernel Module Programming
- Android Application Programming(Activity, Service)
- JNI C/C++ Programming

Follow-up Study Plan - ‘recgraph’ Algorithm





Follow-up Study Plan - ‘recgraph’ Algorithm



Directed maximum spanning tree
on recgraph excluding 1 (source node)
optimal $O((V+E)\log E)$ *(subquadratic)*

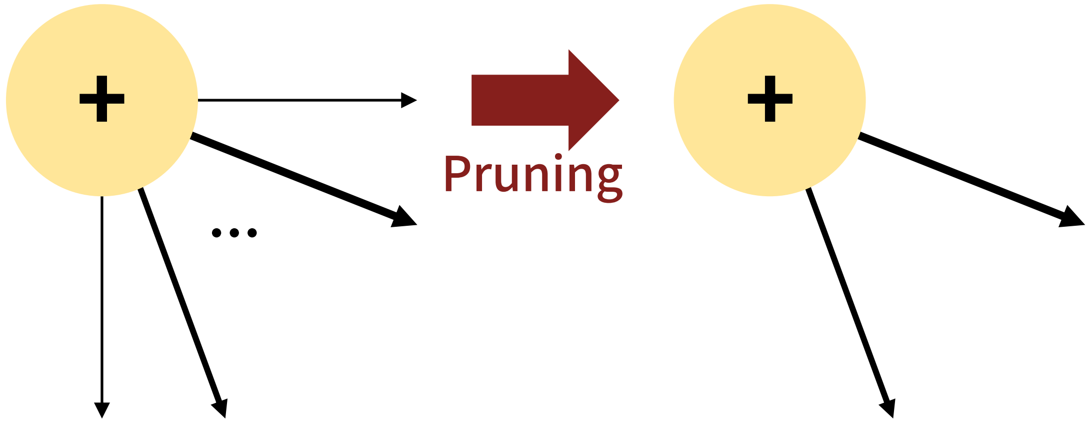
Follow-up Study Plan - ‘recgraph’ Algorithm

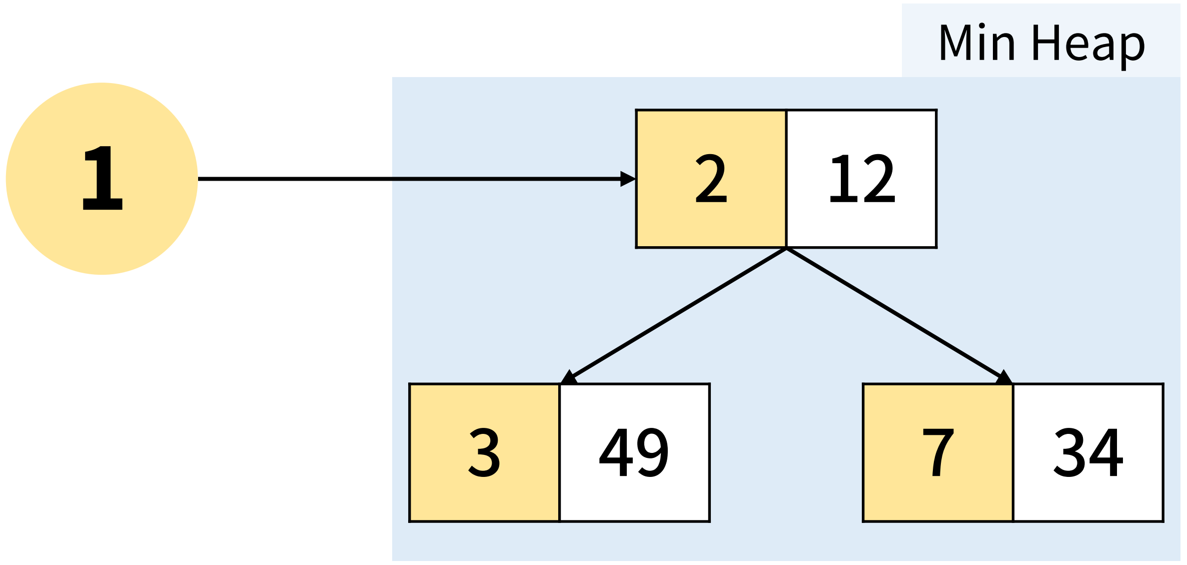
Determining weights (Heuristic)

Multiple continuous additions within a short period of time
No continuous addition again after they have been continuous once
Repeated additions after a particular cluster(long-term memory)
…

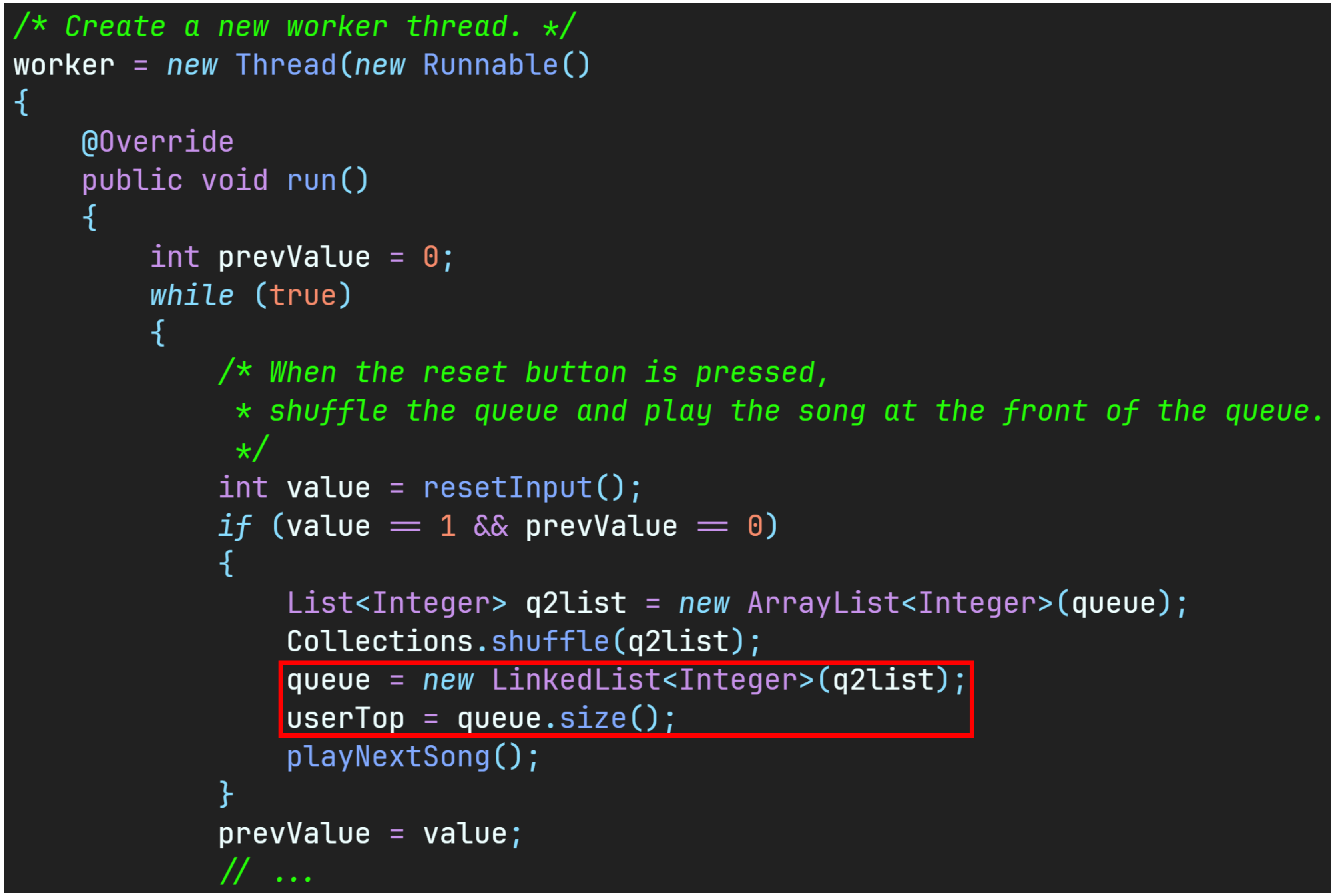
Deep Learning

Follow-up Study Plan - ‘recgraph’ Algorithm





Follow-up Study Plan - Synchronization



```
/* Create a new worker thread. */
worker = new Thread(new Runnable()
{
    @Override
    public void run()
    {
        int prevValue = 0;
        while (true)
        {
            /* When the reset button is pressed,
             * shuffle the queue and play the song at the front of the queue.
             */
            int value = resetInput();
            if (value == 1 && prevValue == 0)
            {
                List<Integer> q2list = new ArrayList<Integer>(queue);
                Collections.shuffle(q2list);
                queue = new LinkedList<Integer>(q2list);
                userTop = queue.size();
                playNextSong();
            }
            prevValue = value;
            // ...
```

Follow-up Study Plan - Application UI

ref: Spotify

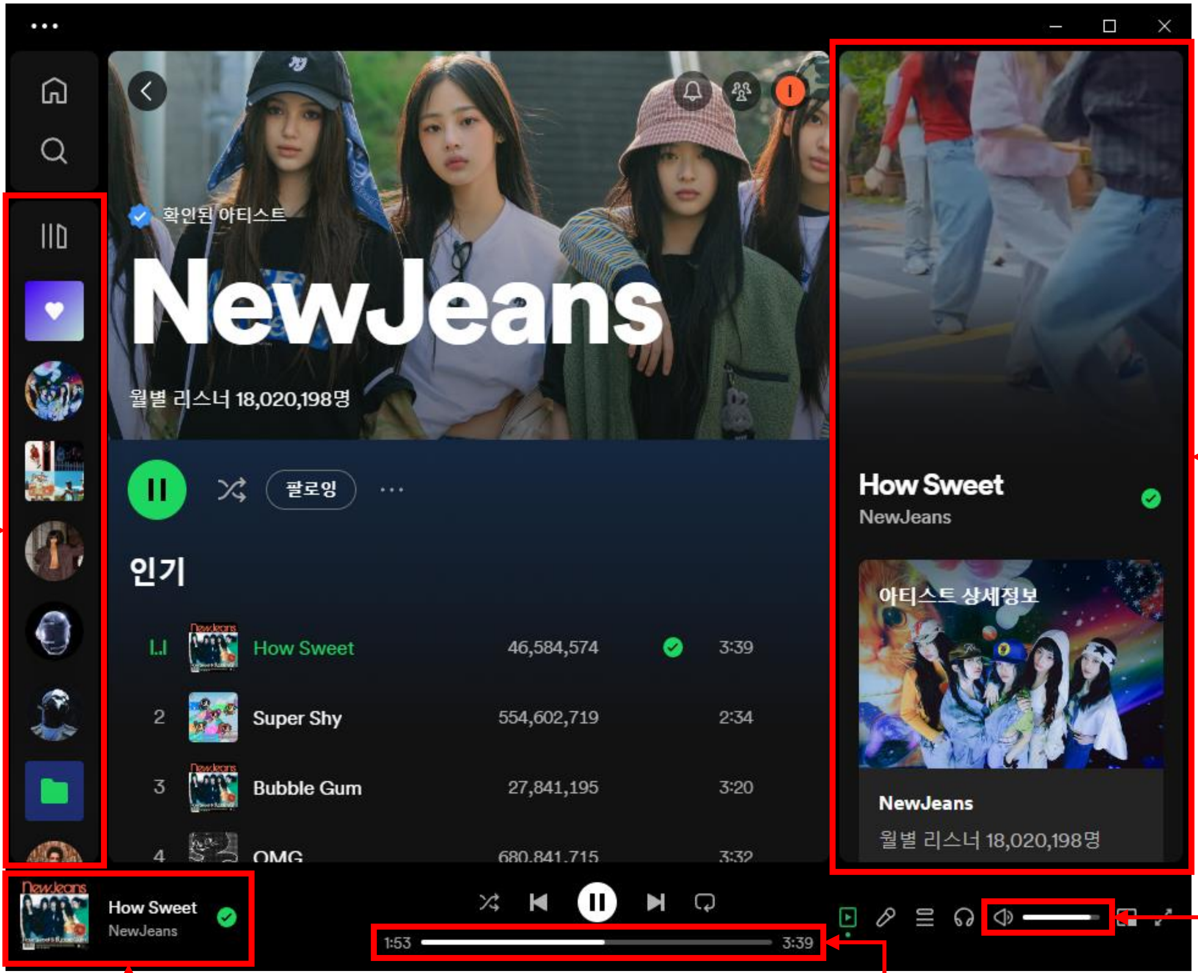
Saved playlists

Music information

Volume bar

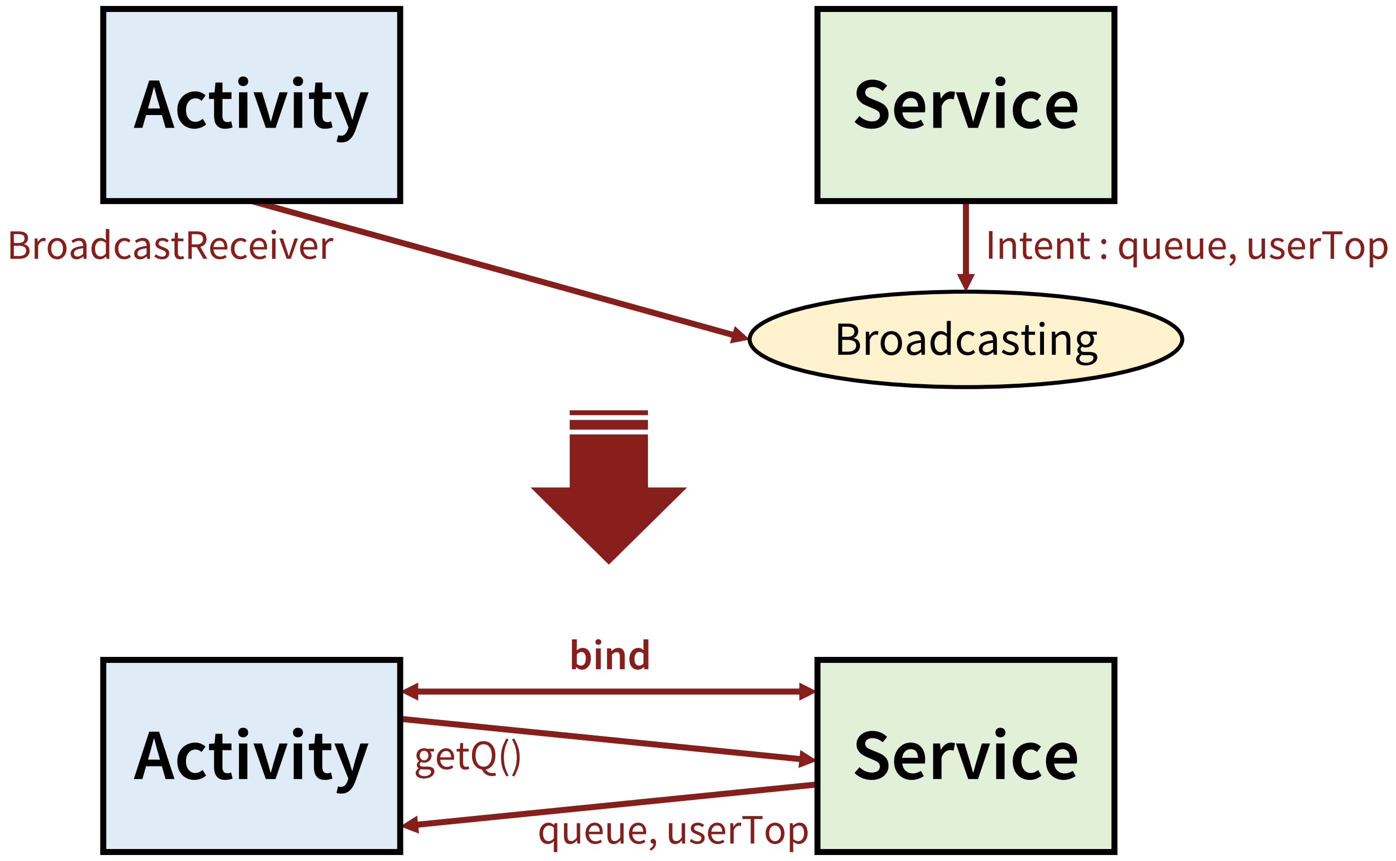
Currently playing music

Progress bar



1	How Sweet	46,584,574	3:39
2	Super Shy	554,602,719	2:34
3	Bubble Gum	27,841,195	3:20
4	OMG	680,841,715	3:32

Follow-up Study Plan - Remote/Bound Service



Thank you for listening.

20211584 Junyeong Jang
Dept. of CS&E, Sogang Univ.

한 학기 동안 수고 많으셨습니다.