NFC Tag Reading and Writing Application Based on Unity3D

Chuan-hong Zhou [a], Wu-xia Bai [b*], Zhen-yang Jiang [c], Kun Yao [d]

149 Yanchang Road, ZhaBei District, Shanghai University, Shanghai, China

[a]chhzhou@staff.shu.edu.cn, [b]18321775027@139.com,

[c]hawkjiang@shu.edu.cn, [d]18321733305@163.com

Keywords: NFC; Unity3D; Cross-platform application transplant; Android system.

Abstract. NFC is a short-range communication of non-contact, combining NFC and smart phone in the field of mobile payments has broad application prospects. Unity3D is the most convenient and reliable software to realize cross-platform application transplant in the current field of game development. Unity3D-based NFC application development innovatively applies Unity3D to industrial sectors, which would greatly promote the development of NFC cross-platform applications. The paper would focus on the implementation architecture of Unity3D-based NFC applications on the Android system. The key points of implementation techniques are described in detail. An application example is also given in the end.

Introduction

NFC (Near Field Communication) is based on RFID theory which has been widely used, but has some unique advantage [1]. First NFC not only realizes the information read and judge but also emphasizes the interaction of information. Secondly, its transmission distance is about 10cm, which is a close private communication. Furthermore, its communication between devices does not need pairing, and compared with the fastest infrared communication, it starts faster nearly 5 times. Besides, NFC is also compatible with the widely used contactless smart card infrastructure [2,3]. These advantages make more and more manufacturers support NFC technology as an electronic wallet technology with wireless transmission environment [4]. Typically, the worldwide growth of mobile technology has enabled many companies to take advantage of its application as complementary sales tools [5] and its features to provide security has also been evaluated [6].

Unity3D is an excellent game engine with graphics, sound, physics and other functions, which has widely used in many fields like smart SNG and modeling and simulation . Besides, it is also a mature cross-platform game engine with supporting such as PC, Linux, iOS, Android, Xbox360 and other mainstream platforms [7]. By Unity3D, application migration between the platforms only needs minimal changes. Unity3D uses C#, JavaScript and other high-level languages to achieve game script functions, which enables developers quickly to develop high-performance applications without understanding the underlying complexity technology [8]. With the diversification of Application Platform and a lot of popularity of iOS, Android phone, Unity3D powerful cross-platform migration capabilities will not be limited to the game field, in a wide range of industries it will also be fully applied.

NFC technology

NFC tag reading and writing application. NFC devices typically include NFC smart devices (e.g. smart phones), NFC reader and NFC tag. NFC tag reading and writing application is interactive communication between NFC smart device and NFC tag.

NFC tag is using for exchanging small data in NFC communication, which could store information like URL, phone number or other text information. NFC Forum defines four different types of tag, namely Type1, Type2, Type3 and Type4. The interactive communication between NFC tags and NFC smart device is NDEF data format which is defined as NFC Data Exchange Format by NFC Forum. Since NDEF already encapsulate the details of the type of NFC tags, the application need not be concerned about communication with what kind of tag [9,10].

Unity3D software

Unity3D cross-platform application porting. Application porting for different platforms have different docking methods which have three basic steps in common [7].

Unity3D-Android Application. As Unity3D generally cannot communicate directly with the system API code, we need to prepare the system plug for the U3D to indirectly call the underlying system API. In Android system, for example, write the appropriate function of the Android system in library file form by eclipse, then introduce the library file into Unity3D. The library file is the Android system plug [7]. The following section describes Specific steps.

Create Android system plug. The steps are as follow.

- Create an Android library project in eclipse, and pay attention to the Package Name and Activity Name. Package Name need to be bound with U3D taking com.project.etagrw as default name. Activity Name will refer to U3D taking MainActivity as default name.
- After creating the project,add classes.jar that is a library file prepared for Android platform by Unty3D to the eclipse project.
- Android underlying application would be completed in MainActivity.java, and the MainActivity must inherit from UnityPlayerActivity, in that way, the method in MainActivity class can be called by script in Unity3D.
- After completing Android underlying application, generate engineering library file taking etagrw.jar as default name which is Android system plug.

Import Android system plug to Unity3D.

- Create a project directory-*Plugins\ Assets* in the Unity3D project, and put the Android system plug into that directory.
- *AndroidMainfest.xml* which is in the Unity3D installation directory should also be placed into *Plugins \ Android* catalog, then modify the node-android:name of the file: *android:name="com.project.“ Package Name”.“Activity Name”"*, where Package Name and Activity Name should be consistent with that are in the Android project, so *android:name=com.project.etagrw.mainActivity* in the *AndroidMainfest.xml*.
- Create a script file in the Unity3D project to communicate with the Android system plugin.
- the Bundle Identifier in the Player Settings window must be consistent with the Package Name in the eclipse project , that is com.project.etagrw

Unity3D-based NFC tag reading and writing application in Android system

Android system introduces NFC API after API9, and currently, Android system offers two packages-*android.nfc* packages and *android.tech* package on NFC development. Unity3D-based NFC tag reading and writing application need to call Android NFC API. Therefore after completing Android system plunge of NFC function in eclipse, introduce it to the Unity3D project. Then, create a C# script file in Unity3D project, in which design application interface and interact with Android system plug.

Analysis of NFC tag reading and writing application. Start the application, and the overall method flow chart is shown in Figure 1.The following section describes Specific steps.

1) The Start method in the C# script used to initialize the C# script and load the Activity that is *MainActivity* is called firstly when Android application started.

2) When the Activity has successfully loaded, onCreat method of the Android plunge should be called to initialize the Android plunge and check the phone NFC Function, and the Update and OnGUI method of C# script are called at each frame(as shown in Fig3.a). Update method is to update the information reading from NFC tag. OnGUI method is to set up and update the information of application UI interface. The function of OnGUI method is shown in Figure 2 , which is interpreted as follows.

- UI interface drawn by OnGUI method consists of five main components including save button, input box, write button, display box and read button. The first three components are writing functional components, and the last two ones are reading functional components.

- The steps of operating writing tag is as follow: Firstly, input the content into the input box, then click the save button to deliver it to the Android plunge(as shown in Fig3.b). Finally, click on the write button to open the reception scheduling of writing tag function in Android plunge.
- The steps of operating reading tag is as follow: Click the read button to open the reception scheduling of reading tag function in Android plunge. The display box would always update the reading content.

3) When the program takes up the main thread, onResume method of Android plunge is called. The reception scheduling of NFC reading and writing function is set in that method. An intent object would be generated with a tag putted in the back of the phone. Whichever reception scheduling is opened, the intent object of NFC tag would be intercepted. If the intent object is successful matched with intent filter, callback method of Android plunge-onNewIntent is called to process corresponding reading or writing function(as shown in Fig3.c&Fig3.d).

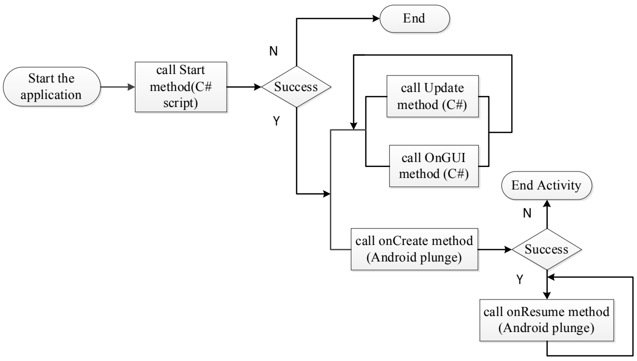

Start the application
call Start method(C# script)
Success
N
Y
End
call Update method (C#)
call OnGUI method (C#)
End Activity
N
call onCreate method (Android plunge)
Success
Y
call onResume method (Android plunge)

Fig.1 The overall method flow chart

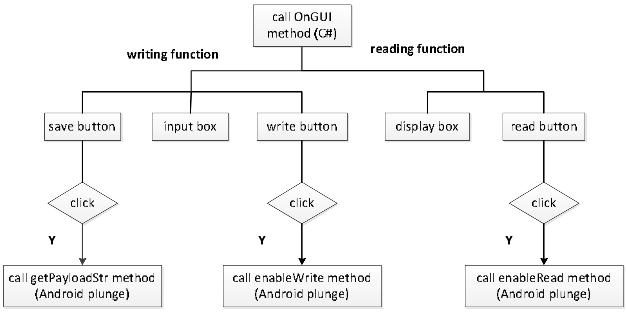
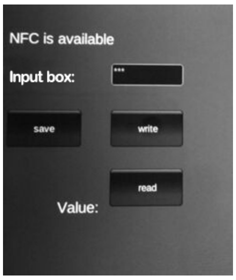


Fig.2 The OnGUI method flow chart

Unity3D-based NFC


NFC is available

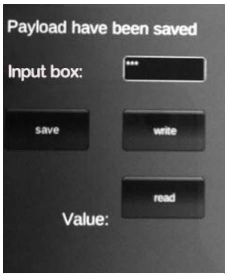
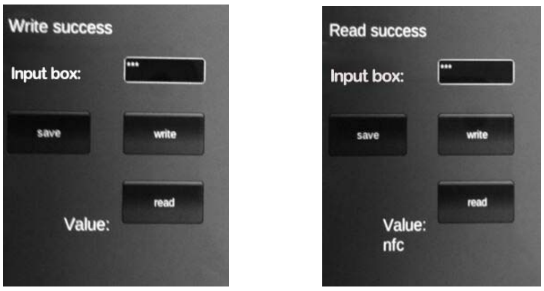

Payload have been saved

Fig3.a The initial interface Fig3.b Clicking save button


Write success

Input box: ***

Value:

Read success

Input box: ***

Value:
nfc

Fig3.c Writing success **Fig3.d Reading success**

Fig.3 The operating interface of the NFC tag reading and writing application

Conclusions

The powerful cross-platform migration capabilities of Unity3D is innovatively applied to the NFC application development. The interaction between Unity3D and the underlying API of the operating system has been researched, and the NFC function of Android system has been achieved by Unity3D, which would be laid the foundation for NFC function transplantation of other platforms.

Acknowledgements This work is supported by subject of Science and Technology Commission of Shanghai Municipality in key technology development and demonstration application of the unmanned factory for industrial robot production(No. 14DZ1100700). We thank Shanghai Key Laboratory of Intelligent Manufacturing and Robotics for assistance with the NFC devices and Unity3D software.

References

[1] Christian Saminger, Michael Stark, Michael Gebhart, Stefan Grünberger, Josef Langer, Introduction of very high bit rates for NFC and RFID, Elektrotechnik and Informationstechnik. 130 (7) 218-223.

[2] Tom Igoe, Don Coleman, Brian Jepson, Beginning NFC: Near Field Communication with Arduino, Android, and PhoneGap, first ed., O'Reilly, New York, 2014.

[3] Vedat Coskun, Busra Ozdenizci, Kerem Ok., A Survey on Near Field Communication (NFC) Technology, Wireless Personal Communications. 71 (3) 2259-2294.

[4] Michael J. Harnisch, Iris Uitz, Near Field Communication (NFC), Informatik-Spektrum, 36 (1) 99-103.

[5] Iviane Ramos-de-Luna, Francisco Montoro-Ríos, Francisco Liébana-Cabanillas, Determinants of the intention to use NFC technology as a payment system: an acceptance model approach, Information Systems and e-Business Management, 2015:284.

[6] Michael Roland, Josef Langer, Comparison of the usability and security of NFC's different operating modes in mobile devices, Elektrotechnik and Informationstechnik. 130 (7): 201-206.

[7] Xizeng Jin, The development of cellphone game using Unity3D, Peking, 2013(In Chinese).

[8] Jingming, Xie, Research on key technologies base Unity3D game engine. ICCSE 2012 - Proceedings of 2012 7th International Conference on Computer Science and Education, Melbourne, VIC, Australia. ICCSE, China, p.695-699(In Chinese).

[9] Bo Zhao, Detailed combat development of NFC Android, Peking University, Peking, 2014.(In Chinese).

[10] Michael Roland, Security Issues in Mobile NFC Devices, first ed, Springer, Cham, 2015.