International Journal of Applied Engineering Research.
ISSN 0973-4562 Volume 10, Number 3 (2015), pp. 7083–7090
© Research India Publications
http://www.ripublication.com/ijaer.htm

Dual Mode Desktop Apps for Windows 8: Solution and Opportunities

Aniket S. Jagtap and Praveen Gubbala
Department of Computer Science and Information Technology,
Symbiosis Institute of Technology,
Gram- Lavale, Tal- Mulshi, Pune, India 412 115
E-mail: aniket.jagtap@sitpune.edu.in, praveeng@sitpune.edu.in

Abstract

Due to switching overhead to launch an application from unlike user environments of Windows 8, there was a need to enforce a new scheme. This paper introduces a scheme so that one can launch an application in respective mode by sensing current user environment. It avoids shifting between two modes to launch an application. Proposed algorithm suggests a solution to design an application against such a limitation.

AMS subject classification:
Keywords: Windows 8, Metro Mode, Dual Mode, WinRT.

1. Introduction

Microsoft Windows is series of Graphical Interface Operating System developed by Microsoft. Windows NT is family of Operating System which has main goal to design hardware and software portability. It is a high level language based, multi-processing, multi-user, processor-independent operating system. Windows 2000, Windows XP, Windows Vista, Windows Server 2003, Windows Server 2008, Windows 7 and Windows 8 are based on Windows NT. Windows 8 is an operating system designed and developed by Microsoft as a part of this family. It introduces various changes to the operating systems platform and also with user experience so as to improve its user experience on tablets and personal computers. It has two modes as part of operating system namely Windows 8 Metro mode which provides immersive user experience and Windows 8 Desktop mode which is traditional Windows Desktop environment. WinRT is Microsofts new execution layer for accessing and leveraging the system resources and programming model to develop the metro apps.

Table 1: Operating Systems and their features.

Features	**Windows 7**	**Windows 8**
Availability of Desktop Mode	Yes	Yes
Availability of Metro Mode	No	Yes
Support for Win32 APIs	Yes	Yes
Support for WinRT APIs	No	Yes
Support for Immersive User Experience	No	Yes
Support for Touch enable User Interface	Yes	Yes

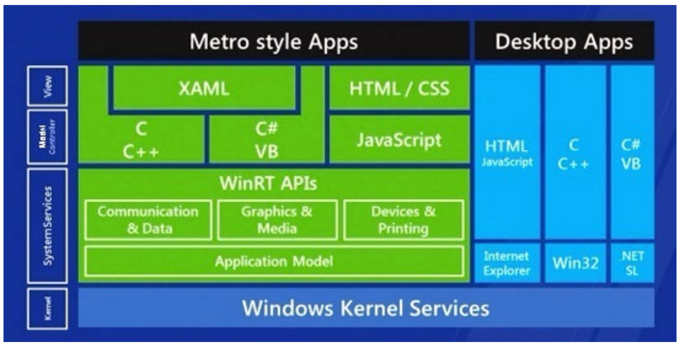


Figure 1: Windows 8 Platforms and Tools. [2]

Table 1 describes new supporting technologies of Windows 8 over Windows 7.

Figure 1 illustrates Windows 8 platform, WinRT and Win32 APIs, and supported technologies by respective API. WinRT apps often known as Windows Store Apps. Win32 was introduced with Windows NT. It is the 32-bit API for modern versions of Windows. WinRT is Windows Runtime which replaces the old Win32 libraries to access system resources. It is a set of APIs which is used to build metro applications on Windows 8. It is a new programming model introduced by Microsoft as execution layer to develop metro software which can be running on tablets and personal computers. The advantages of WinRT APIs over Win32 APIs are as follows,

- Supports the ARM as well as x86, x64 based CPU architecture
- Simpler and more stable API than old Win32 API
- Easy access to system's hardware resources
- Applications created are more secure,safe and supports sandboxed security model
- Immersive user interface
- Supports Multi-touch UI.

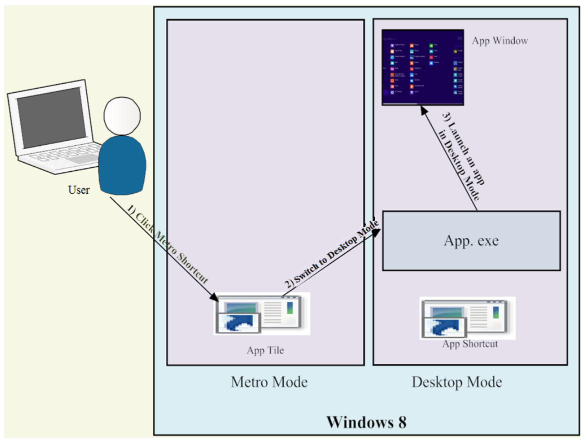


Figure 2: Traditional Method to launch an application.

Application deployment in Windows 8 Metro vs. Desktop

Microsoft introduces Windows Store which is an online store to distribute new metro apps among the end user. So to distribute metro app is necessary to upload it to Windows store by validating the application package using Windows App Certification Kit. Another way to distribute Windows Store app within the enterprise by using Side-loading product keys activation which are provided by Microsoft [6]. While desktop application can be distributed via web downloads, network share etc. It can be deployed via existing desktop application methods like MSI packaged, click-one installer and .zip extraction with file copy. All executable must be digitally signed.

2. Related Work

Windows Store App runs under app container and uses WinRT APIs. These apps is packaged as .appx file and uses Windows Store for distribution. These apps do not have backward compatibility to previous platforms of Windows such as Windows 7.

Windows 8 desktop apps are traditional Windows desktops apps like applications in windows 7. Which can be launched by user using interaction with metro tile of an application, if it is not available in metro mode then it will launch in desktop mode, as illustrated in Figure 2.

3. Design

The New User Experience Desktop Browsers [2] are desktop web browsers but can have the capability to launch and run on Windows 8 Desktop as well as Metro mode. Microsoft allows this capability to internet browsers which are the default viewer for http:// protocol on x86, x64 architecture [1]. Proposed scheme extends the mechanism to launch an application in respective user environment.

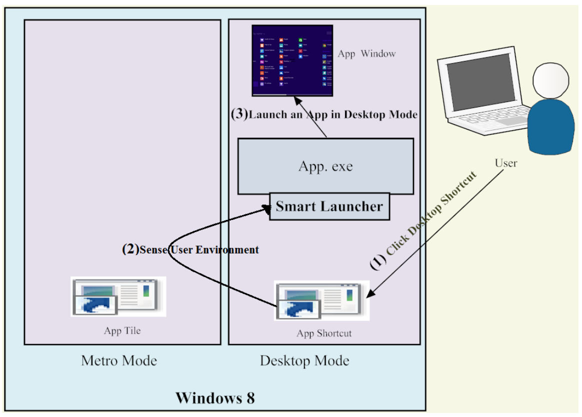


Figure 3: Method to launch Dual Mode App in Desktop Mode.

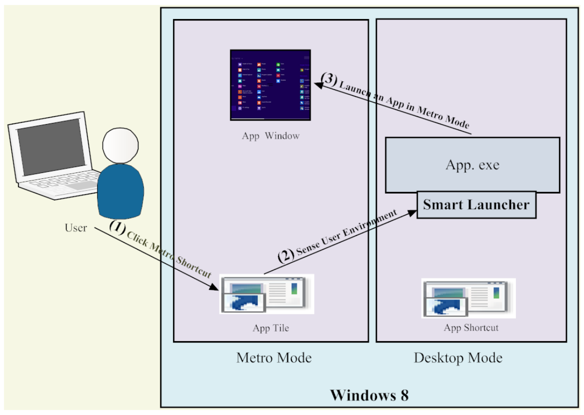


Figure 4: Method to launch Dual Mode App in Metro Mode.

Figure 3, illustrates the approach to launch Dual mode app, if an user interacts with the metro tile then it will sense current user environment as Metro and will launch an application in metro mode automatically. Same will be applicable if user interacts with desktop shortcut of Dual mode app, Which is shown in Figure 4.The main contribution of this paper is as follows.

The current study is to extend the capability of browser to normal desktop applications by which one application can have ability to launch on both modes of Windows 8.

The algorithm and flowchart provides the detailed information about the desktop application which has capability to launch automatically by sensing current user environment and run it on respective environment. Prerequisites are an application executable must be digitally signed and create start shortcut for an application.

Step 1: Interaction with launch points.

Step 2: Resolve App_User_Model_Id to application.

Step 3: If App_User_Model_Id set to dual mode property then goto step 4.
Else set App_User_Model_Id to dual mode property.

Step 4: Delegate the Execution of an application.

Step 5: If Application set to HTTP protocol default then goto step 6.
Else set application to HTTP protocol default.

Step 6: Determine the current user environment.

Step 7: If current user environment runs in low integrity level then set environment value: = Immersive mode and goto step 8.
Else set environment value: = Desktop mode goto step 9.

Step 8: Launch an application in Windows 8 metro mode.

Step 9: Launch an application in Windows 8 desktop mode.

From given algorithm and flowchart,

- The launch points that may be Windows 8 desktop shortcut for an application and in metro mode it is metro tiles.
- App_User_Modelid is an Application User Model IDs used by system to uniquely recognize processes, files regarding specific application. For the dual mode applications App_User_Modelid must be set to true for dual mode property.
- Here delegate the execution launching an application till determining the current user environment.
- As the dual mode is only available with new user experience internet browsers and we are extending the capabilities of internet browser so an application must set default to http:// protocol [1].
- Applications running in desktop mode are basically run in high or medium integrity level where as metro style application run in low integrity level of Windows 8. Depending on the integrity level mechanism it is easy to determine the current user environment.
- By determining the appropriate values, now it is easy for a system to make decision that where to launch an application that is in metro mode or in desktop mode.

There are number of areas of Windows 8 system where you must use WinRT APIs to access the system resources for Windows 8 metro apps, but no Win32 equivalent API included in WinRT. Windows Store Apps can use a subset of Win32 and COM API. This is the case where you have to make use of Win32 shared library which can be accessed by making use of WIN4API_FAMILY, WINAPI_FAMILY_PARTITION macros [3, 4].

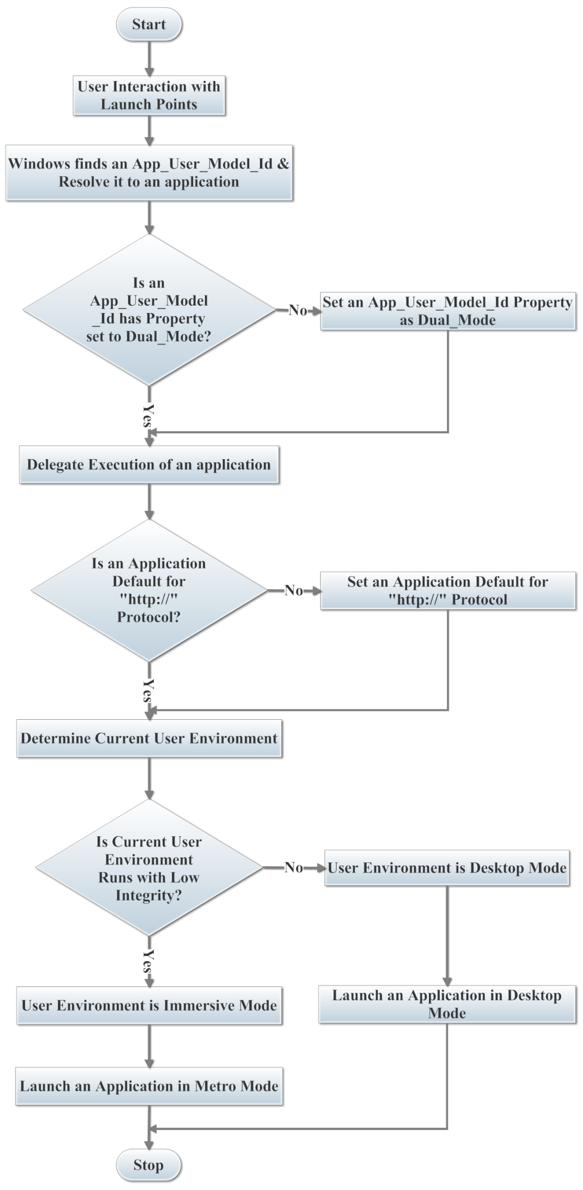

Start
User Interaction with Launch Points
Windows finds an App_User_Model_Id & Resolve it to an application
Is an App_User_Model_Id has Property set to Dual_Mode?
No
Set an App_User_Model_Id Property as Dual_Mode
Yes
Delegate Execution of an application
Is an Application Default for "http://" Protocol?
No
Set an Application Default for "http://" Protocol
Yes
Determine Current User Environment
Is Current User Environment Runs with Low Integrity?
No
User Environment is Desktop Mode
Yes
User Environment is Immersive Mode
Launch an Application in Desktop Mode
Launch an Application in Metro Mode
Stop

Figure 5: Flow chart for proposed system.

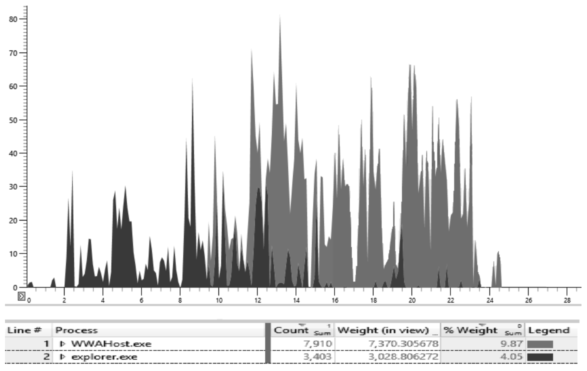
The Windows App Certification Kit ensures that an application uses only APIs belongs to these subset. The proposed system can use these shared library Win32 APIs and DirectX for the UI design [4, 5]. The flow chart determines the current user environment and accordingly launches an application in previously determined user environment that are Windows 8 metro or Desktop. By extending the capability of New User Experience Desktop browsers [1], it is possible to develop one desktop application but an application can have capability to launch and run on both the mode of Windows 8.

4. Results and Discussion

Table 2: Devices, OS and supported Windows API.

Device	Operating System	Supported Windows API
Personal Computer	Windows 8	Win32 and WinRT
Tablet	Windows RT	WinRT
Smart Phone	Windows Phone 8	Windows Phone Runtime

From table 2, this dual mode desktop application can run only on personal computers because the shared library have a set of the Win32 API and those API only supported in Windows 8 in the personal computer.



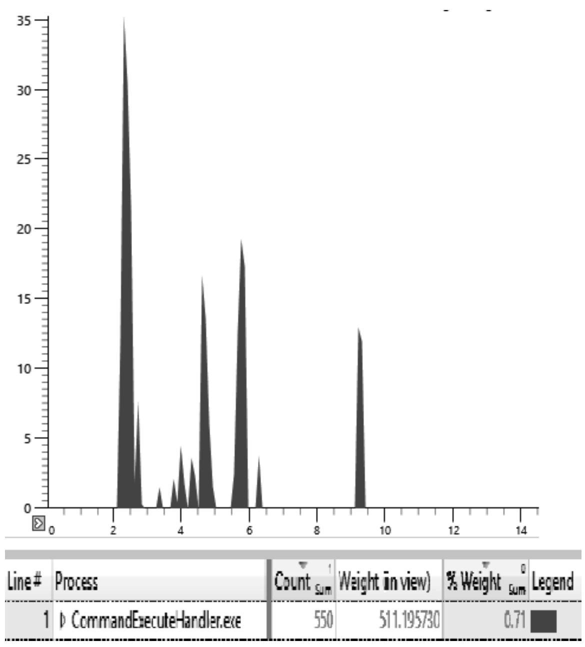
Line #	Process	Count Sum	Weight (in view)	% Weight Sum	Legend
1	▷ WWAHost.exe	7,910	7,370.305678	9.87	▨
2	▷ explorer.exe	3,403	3,028.806272	4.05	■

Figure 6: Graph of CPU Usage while switching between the modes of Windows 8.

The metrics used to perform analysis are CPU Usage vs. Time in seconds. Graph shown in Figure 6, WWWAHost.exe is a process that hosts Windows Store app to run in metro mode of Windows 8 and explorer.exe is a process in desktop mode. While switching between the modes these two processes get launched, CPU usages given in % weight for those processes are 9.87 and 4.05 respectively. But graph shown in Figure 7, CommandExecuteHandler.exe is the dual mode desktop app which has CPU usage in % weight as 0.71. Thus, Dual mode desktop reduces the CPU usage while launching an app as compared to the CPU usage for switching between two modes of Windows 8.

5. Conclusion

The app can work automatically by sensing current user environment. It launches an application according to current user environment, which increases availability of an application in both the modes. Also it eliminates unnecessary switching between both the modes of windows 8 to interact with the applications. Further design and development may vary with different types of applications but the given algorithm provides the basic functionality of sensing user environment and gathering the appropriate parameters from the operating system to launch particular an application in respective mode.



Line#	Process	Count Sum	Weight (in view)	% Weight Sum	Legend
1	▷ CommandExecuteHandler.exe	550	511.195730	0.71	■

Figure 7: Graph of CPU usage while launching Dual mode in Windows 8.

Thus, proposed algorithm provides the capability to launch an application on Windows 8 metro as well as desktop mod by eliminating unnecessary switching between the Windows 8 modes.

References

[1] Microsoft. Developing a new experience enabled desktop browser. September 17, 2013. http://go.microsoft.com/fwlink/p/?linkid=243079 (Accessed October 2, 2012)

[2] Michael, Mayberry (2012). WinRT Revealed. New York City: Apress.

[3] Chuck Walbourn-MSFT. Dual-use Coding Technique for Games, Part1. September 17, 2012. http://blogs.msdn.com/b/chuckw/archive/2012/09/17/dual-use-coding-techniques-for-games.aspx (Accessed November 04, 2013)

[4] Chuck Walbourn-MSFT. Dual-use Coding Technique for Games, Part2. September 17, 2012. http://blogs.msdn.com/b/chuckw/archive/2012/09/18/dual-use-coding-techniques-for-games-part-2.aspx (Accessed November 04, 2013)

[5] Chuck Walbourn-MSFT. Dual-use Coding Technique for Games, Part2. September 17, 2012. http://blogs.msdn.com/b/chuckw/archive/2012/09/18/dual-use-coding-techniques-for-games-part-3.aspx (Accessed November 04, 2013)

[6] Rey Fleming. Windows 8 in Education: Windows Store App and Deployment. September 02, 2013. http://blogs.msdn.com/b/education/archive/2013/09/03/windows-8-in-education-windows-store-apps-and-deployment.aspx (Accessed October 15, 2013)