Asset Standards

It is possible to use your own original files for all kinds of assets such as images and audio.
By copying the asset file into the appropriate folder you can select the asset from the editor.

- Image Assets
- Tileset Details
- Audio Assets
- Movie Assets

Image Assets

All images will use the PNG format.

Animation Patterns (img/animations)

These are primarily the images used for animations shown as effects in the Battle Screen.
1 cell contains an image that measures 192x192 in size, and 5 cells lined up horizontally are treated as a block which will be stretched vertically only as necessary, becoming 1 file. A file can contain up to 20 blocks (100 cells).
* Images used are the same size as in the previous version (VX Ace).

Battlebacks (img/battlebacks1, img/battlebacks2)

Images used as the backgrounds for the Battle Screen.
A battleback measures 1000x740 in size, backgrounds in the battlebacks1 folder are primarily for floors, and those in the battlebacks2 folder are primarily for walls, and you can put these into any combination that you like in order to create your battleback.
The upper part of images in front-view battles and the lower part of images in side-view battles will be cropped then displayed.

Characters (img/characters)

Images shown for the characters on the map.
The size for a character image can be changed freely (normally 48x48), and consist of 4 directions (down, left, right, up) and 3 patterns for a total of 12 patterns that will be arranged in the provided order. A file for 1 character will be arranged into 2 rows and 4 columns equaling 8 images. The size of the character will be calculated using 1/12 of the width and 1/8 of the height of this file. Moreover, characters will be shown 6 pixels above tile so that they appear more natural when on top of buildings.

- It is possible treat 1 character as 1 file by including "$" at the beginning of the file name.

- Including a "!" at the beginning of the file name will prevent the image from being shifted 6 pixels and will no longer appear half transparent by bush elements. This is primarily used for objects found in the map such as doors and treasure chests.

- It is also possible to use the "$" and "!" special characters together.

Front-view Enemies (img/enemies)

Images for enemies that are displayed in front-view battle. You can choose any size you want for the image.

Face (img/faces)

Images which are displayed in the Menu Screen and message windows.
Face images are 144x144 in size, and arranged into 2 rows and 4 columns equaling 8 images.

Parallax (img/parallaxes)

Images shown behind maps. You can choose any size you want for the image.
Like wallpaper, the top, bottom, left and right sides of the image will be connected when wanting to loop the image.
Additionally, parallax files will be treated as floors (no parallax) when there is a "!" at the beginning of the file name.

Pictures (img/pictures)

Images which are displayed by using event commands during the game. You can choose any size you want for the image.

Side-view Allies (img/sv_actors)

Actor images which are displayed in side-view battle.
Please refer to [Side-view Character Standards].

Side-view Enemies (img/sv_enemies)

Images for enemies that are displayed in side-view battle. You can choose any size you want for the image.

System Images (img/system)

The various images used in the overall game.

Balloon.png

Asset used for actor chat balloons. The transparent area below the image is defined by the user.

ButtonSet.png

Buttons used when the game is displayed on devices such as smartphones which support touch controls. This can be used for changing things like the number of items when purchasing them.

Damage.png

Asset used for displaying damage pop-ups in the Battle Screen.

GameOver.png

Asset displayed on the Game Over Screen.

Loading.png

Asset used for the Loading Screen.

IconSet.png

Assets used for the icons for items, weapons, skills, etc.

Shadow1.png

Asset for the shadow displayed when flying a plane.

Shadow2.png

Asset for the shadows of allies in side-view battle.

States.png

Asset for the states of allies in side-view battle.

Weapons1.png, Weapons2.png, Weapons3.png

The weapon images that are displayed when allies perform normal attacks in side-view battle.
There are 12 different types of images for Weapons1 and Weapons2. Weapons3 has 6 different types of images. Weapons3 is defined by the user.

Window.png

Collection of image assets that compose the windows.

Tilesets (img/tilesets)

Images which consist of the tiles (also known as map chips) that make up a map.

Title Screen (img/titles1, img/titles2)

Images displayed on the Title Screen.
The size for these images is 816x624. Titles1 contains files for the main background, and titles2 contains images for frames, etc. Use these in any combination to create the Title Screen.

1 tile is 48x48 in size, and tiles need to be grouped in the 5 types of sets, A through E, below.

Additionally, the specifications for some tiles can change according to the contents set under [Mode] found in [Tilesets] in the database.

Set A

This set will be used as the lower layer when drawing the map. This set is divided further into 5 parts, with most of them being called [Autotiles], which are composed of special tiles that have their boundary lines automatically created.

Autotiles are, as a rule, arranged in a pattern composed of 6 tiles as seen in the illustration below, making up the basic structure of the tiles.
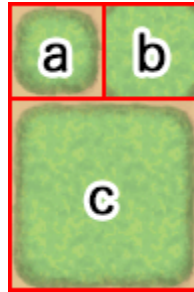
a

    Representative Pattern (for displaying in the tile palette)

b

    Pattern with boundaries at each corner

c

    Group Pattern (refers to group of tiles with one in the center and 1 in each of the 8 directions)

If the autotile located in the (8,8) position from the bottom-right is transparent, that autotile will be evaluated as a "forest type". If a forest tile has the bush element assigned to it, character images will not appear as half transparent in the 8 types of tiles below which includes the bottom right and bottom left boundaries.

Part 1
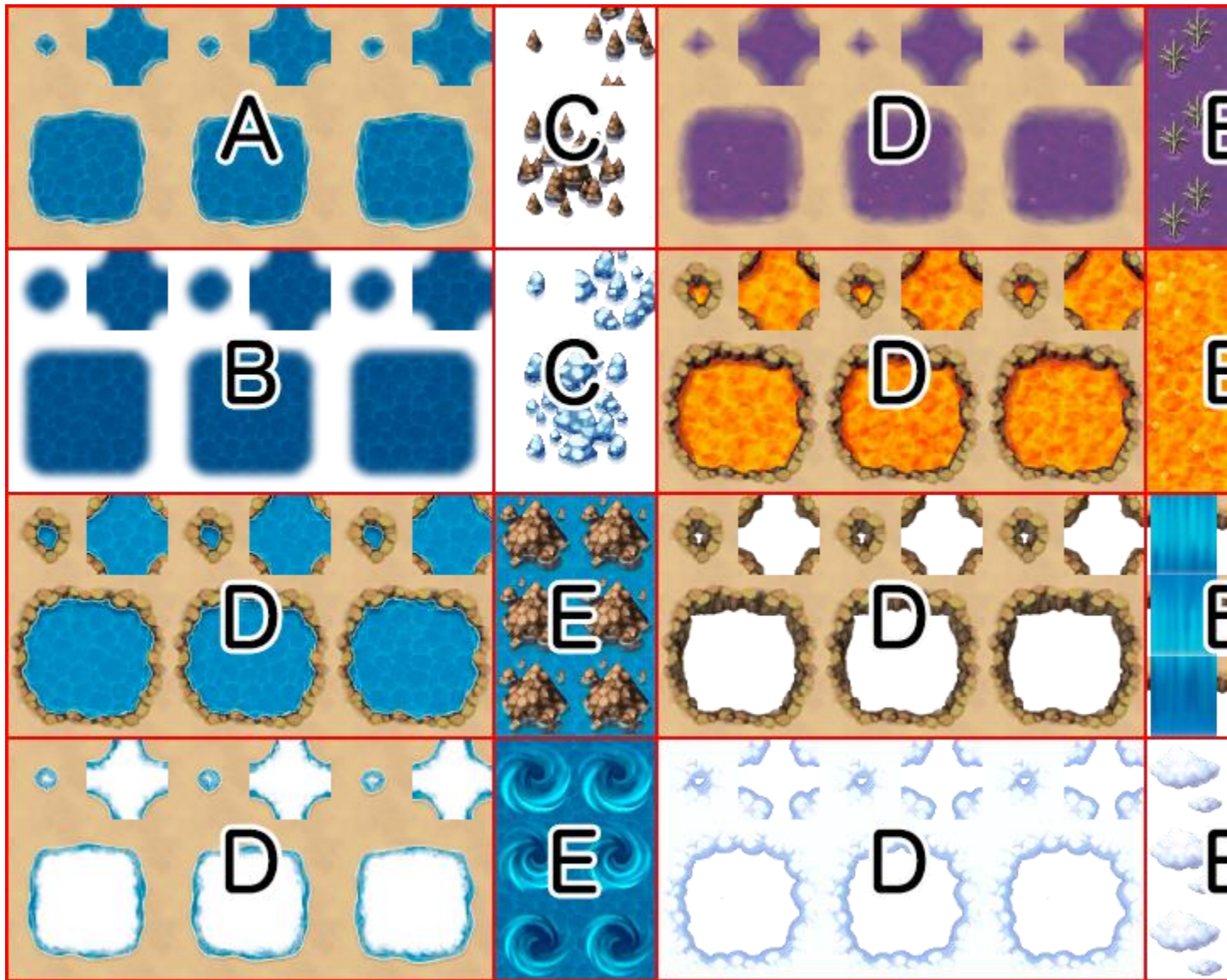
These are 768x576 in size and made up of the 5-pattern blocks as in the illustration above. Basically, tiles in this part will not have a boundary created even if they touch.

Boats and ships can only travel through the tiles in this part. However, tiles in this tileset will no longer be able to be entered using boats and ships if the tileset is configured to allow players to walk on the tiles.

Block A
Autotiles used as ocean tiles. By placing 3 autotile basic patterns horizontally in a row, it is possible to animate them.

Block B
Autotiles used as deep ocean tiles. Boundaries for ocean tiles will be created only when tiles in this block touch tiles in part 1. Tiles in Block A will automatically complete the transparent color of this block. Just like Block A, by placing 3 autotile basic patterns horizontally in a row, it is possible to animate them. Moreover, boats cannot travel through tiles in this block.

Block C
Autotiles which decorate ocean tiles in Block A. Tiles in Block A will automatically complete the transparent color of this block. Additionally, boats and ships cannot travel through tiles in this block.

Block D
Autotiles used as water tiles. By placing 3 autotile basic patterns horizontally in a row, it is possible to animate them.

Block E

Used for waterfall tiles. You can create a group pattern by placing two tiles horizontally, and animate them by placing 3 vertically in a row. Additionally, boats and ships cannot travel through tiles in this block.

Part 2
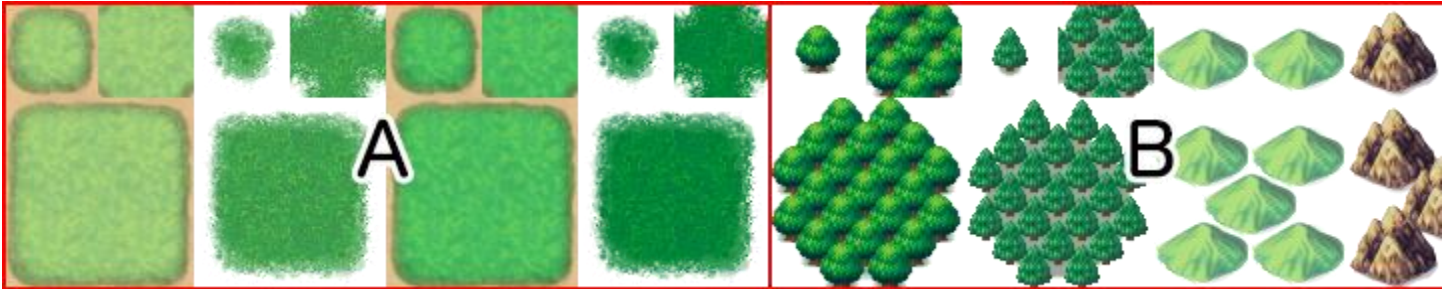
These are 768x576 in size and composed of 4 2-pattern blocks placed vertically in a row as in the illustration above. Specifications for this part only can change according to the contents set under [Mode] found in [Tilesets] in the database.

If the tiles in this part have the counter element, they will be used as autotiles to create tables, and the bottom of the pattern will be displayed as shifted 12 pixels down when placed.

Block A (Field Type)

Composed using 4-pattern autotiles, and will be handled as 1 only, 1 and 2 overlapping, 3 only, 3 and 4 overlapping in the actual tileset.
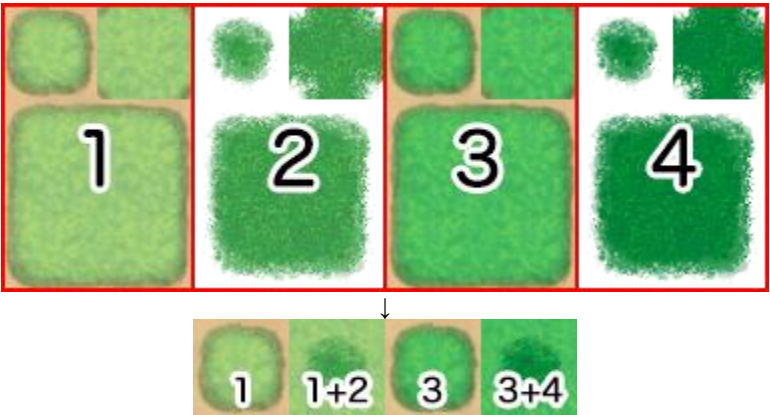
Block B (Field Type)

It is possible to store 4 patterns, and are special tiles that can be placed over tiles in Block A in the actual tileset.

Block A (Area Type)

It is possible to store 4 patterns, and are tiles that can be placed over tiles in Block B in the actual tileset.

Block B (Area Type)

It is possible to store 4 patterns, and are tiles that can be placed over tiles in Block A in the actual tileset.

Part 3

Autotiles which will be primarily used for the appearance of buildings. These are 768x384 in size, and are composed by placing 8 tiles horizontally and 4 tiles vertically, formed using only the autotile group pattern.

By placing two or more tiles in this part together vertically when designing your map, shadows will automatically be created on the adjacent touching tile on the right side. However, shadows will not be automatically generated if the adjacent tile belongs to a part other than Part 2 (excluding Block C) or Part 5.

Part 4

Autotiles which will be primarily used for walls. These are also used for walls for dungeon instances. These are 768x720 in size. Composed by placing 8 tiles horizontally and 3 tiles vertically using autotile basic structures and those tiles placed vertically in a row using only the autotile group pattern.

By placing two or more tiles in this part together vertically when designing your map, shadows will automatically be created on the adjacent touching tile on the right side. However, shadows will not be automatically generated if the adjacent tile belongs to a part other than Part 2 (excluding Block C) or Part 5.

Part 5

These are 384x768 in size and please be sure to place the tiles here in an 8x16 arrangement. Tiles contained in this file will all be treated as normal tiles. The 3rd, 5th and 7th tiles from the top are used also for the floors of dungeon instances.

Set B through Set E

These sets will be used as the upper layers when drawing the map.
These are 768x768 in size and be sure to place the tiles here in a 16x16 arrangement.

- Leave the tile located in the top left of Set B blank as this represents nothing being placed in the upper layer.

Audio Files (Music, Sound Effects)

The file formats that are playable depend on the operating environment, so please prepare the below 2 file formats with the same file name.

- Ogg Vorbis(.ogg)

- AAC(.m4a)

* When creating a game that will be played on Windows or Mac, please prepare only the Ogg Vorbis file format. When creating a game that will be released on the web or made for Android/iOS, both file formats are required.

Video Files (Movies)

The file formats that are playable depend on the operating environment, so please prepare the below 2 file formats with the same file name.

- WebM(.webm)

- MP4 (.mp4 encoded with H.264 codec)

* When creating a game that will be played on Windows or Mac, please prepare only the WebM file format. When creating a game that will be released on the web or made for Android/iOS, both file formats are required.

Side-view Character Standards

These are the standards for images used as the battle graphics of actors displayed in Side-view Mode. It is possible to use one image file per character.

Specifications

Images are stored in a file in a 9x6=54 pattern.
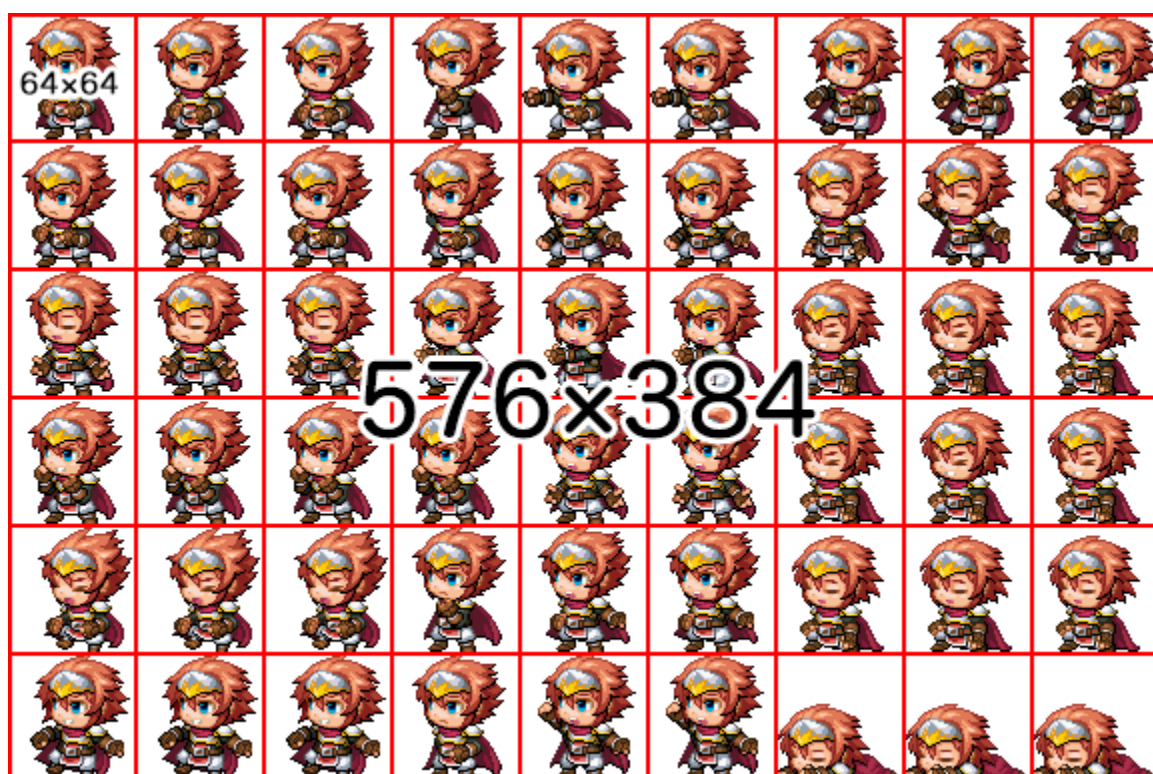
- The width of the image is 9 and the height is 6, which becomes the size for 1 character.

- The image size for standard assets is 576x384.

3 patterns lined up in a row will be treated as one motion.

- Looping standby motions will be repeatedly displayed in a 1-2-3-2 order starting from the left.

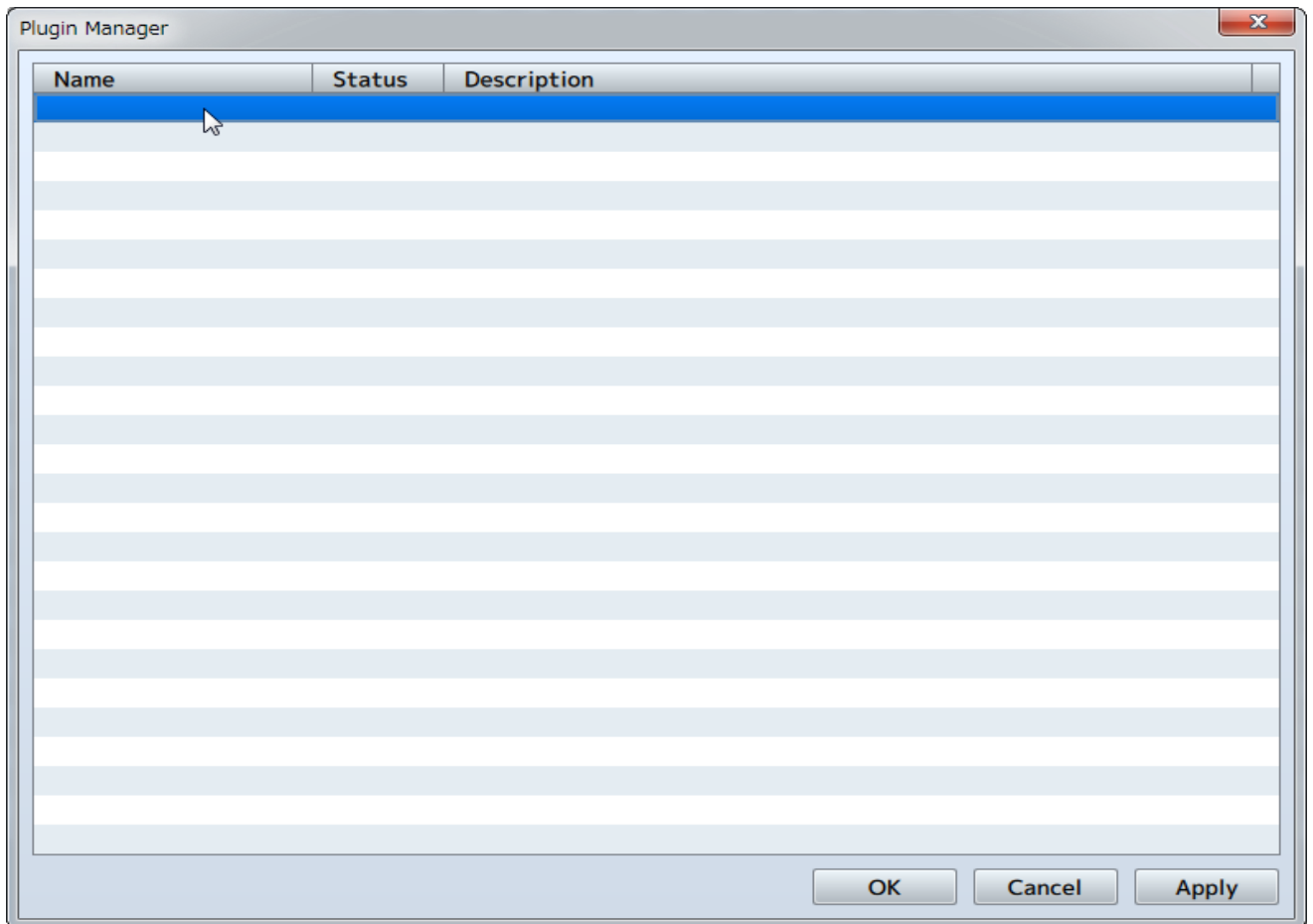- Motions that occur only 1 time such as damage motions will be shown in 1-2-3 order starting from the left.

Enabling Plugins

Users can find easier to use plugin features in RPG Maker MV.

Enabling Plugins

Enable plugins using the [Plugin Manager] found under [Tools].

1. Double-click an empty row.

2. Select the plugin you want to use and click the [OK] button.
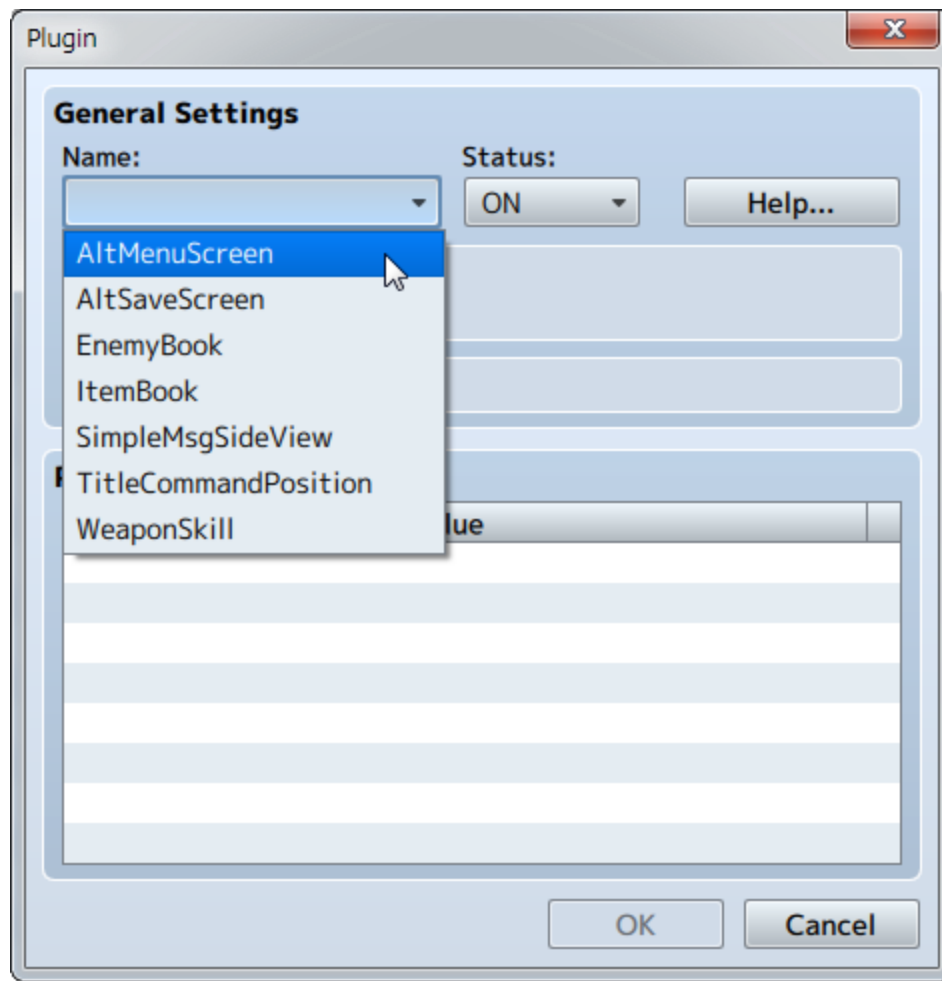
3. The selected plugin will appear in the list.
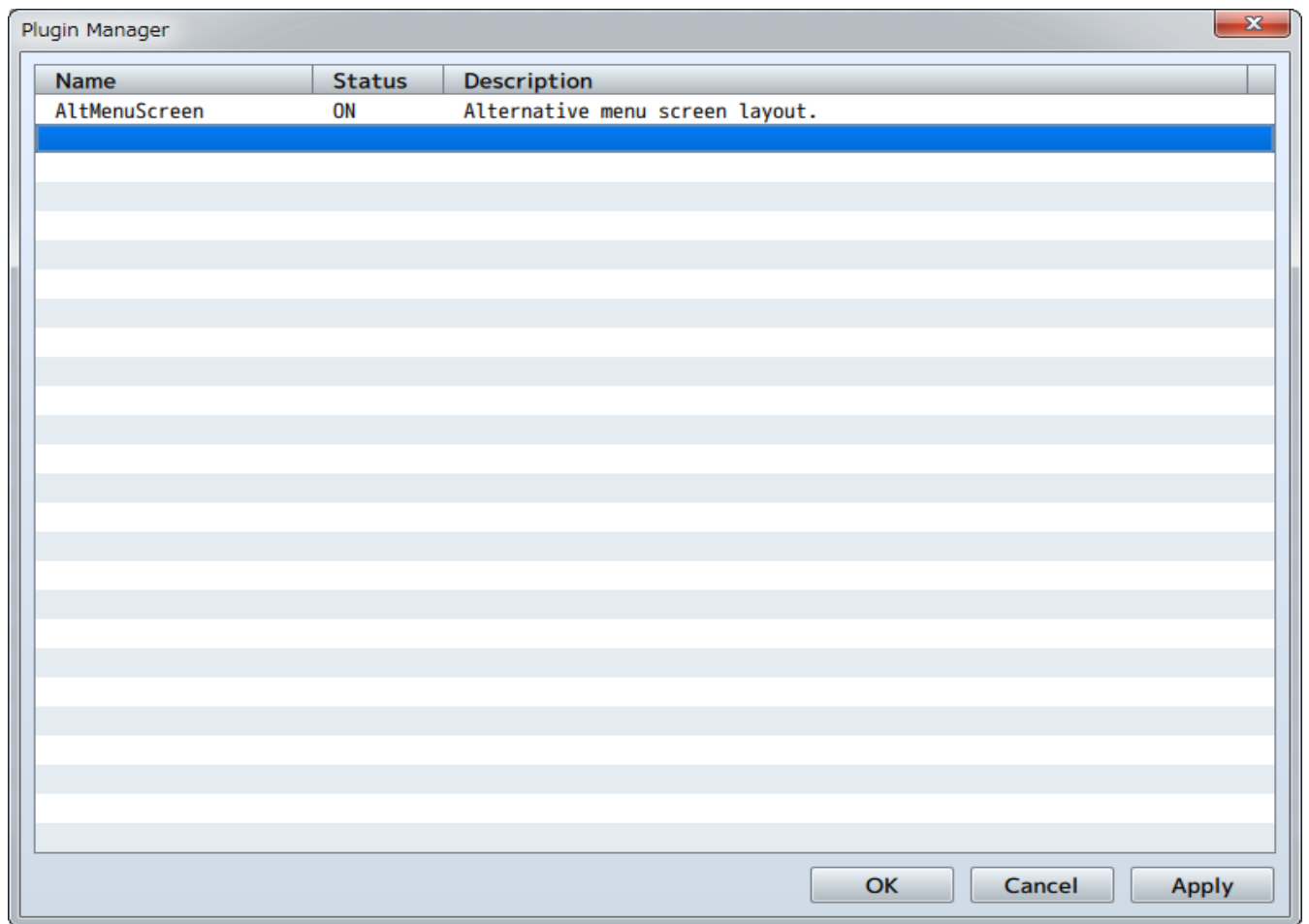
Enabling Unofficial Plugins

When adding a plugin, after copying the desired plugin's js file to the "plugins" folder found in the project folder's "js" folder, enable it using the above procedure.

1. Copy the desired plugin's js file to the "plugins" folder found in the project folder's "js" folder.
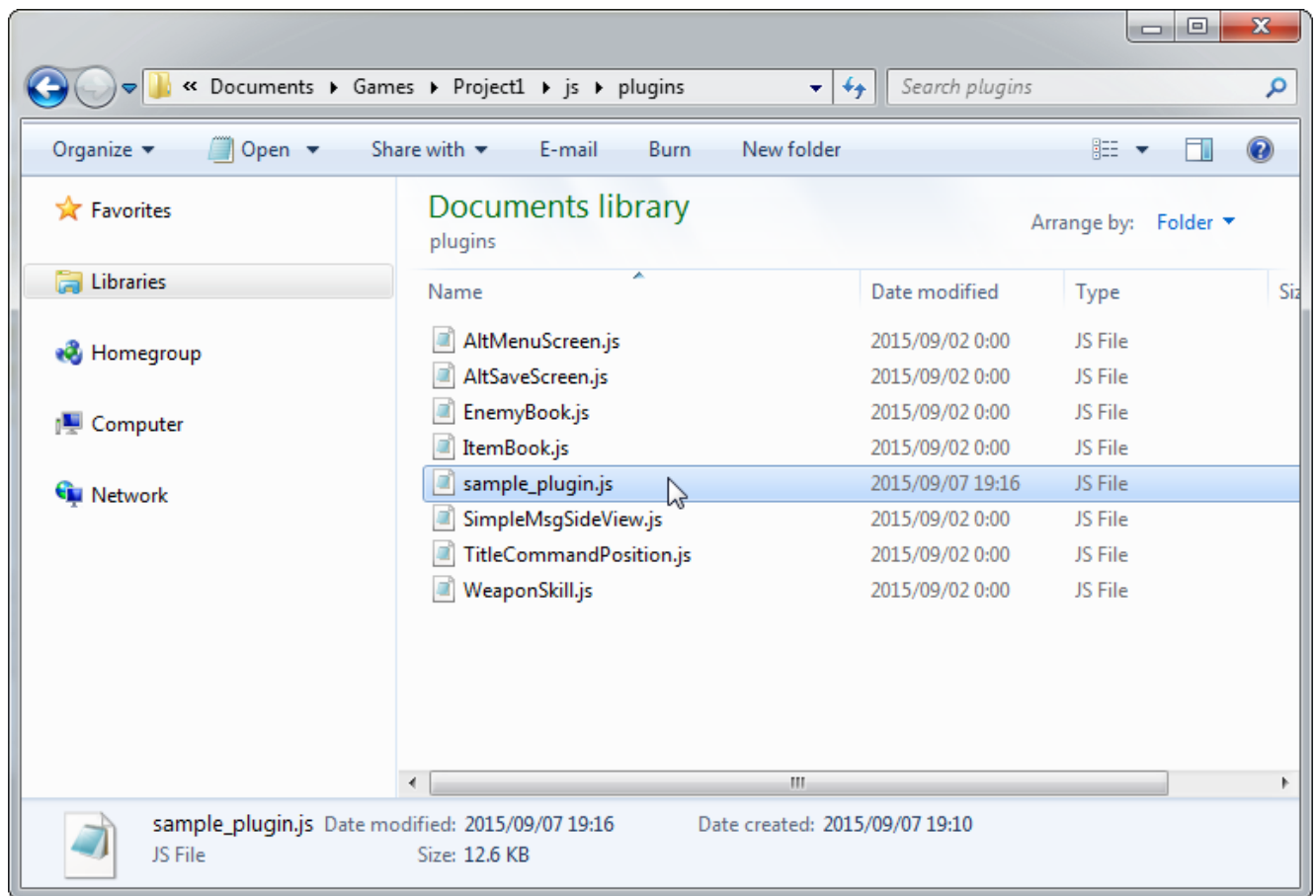
2. Double-click an empty row in the Plugin Manager and the name of the plugin that was added will appear when you click [Name]. Select the name and click [OK].
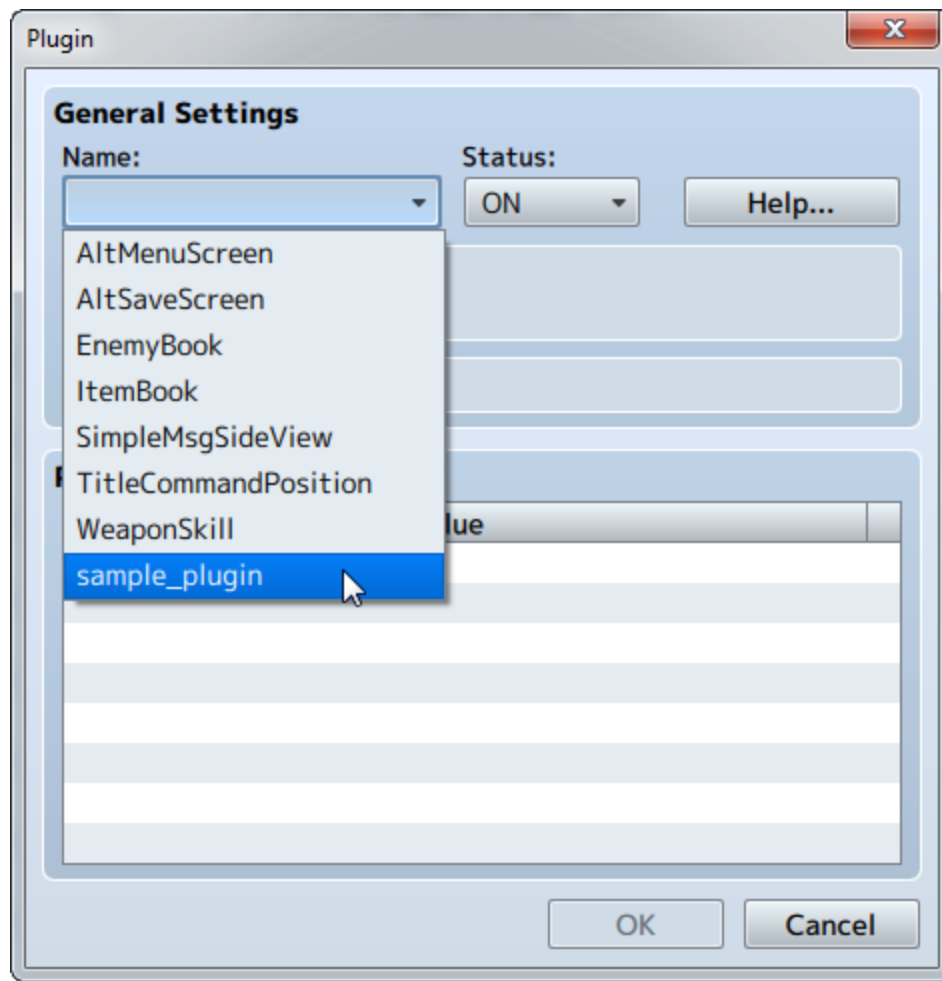
3. The selected plugin will appear in the list.
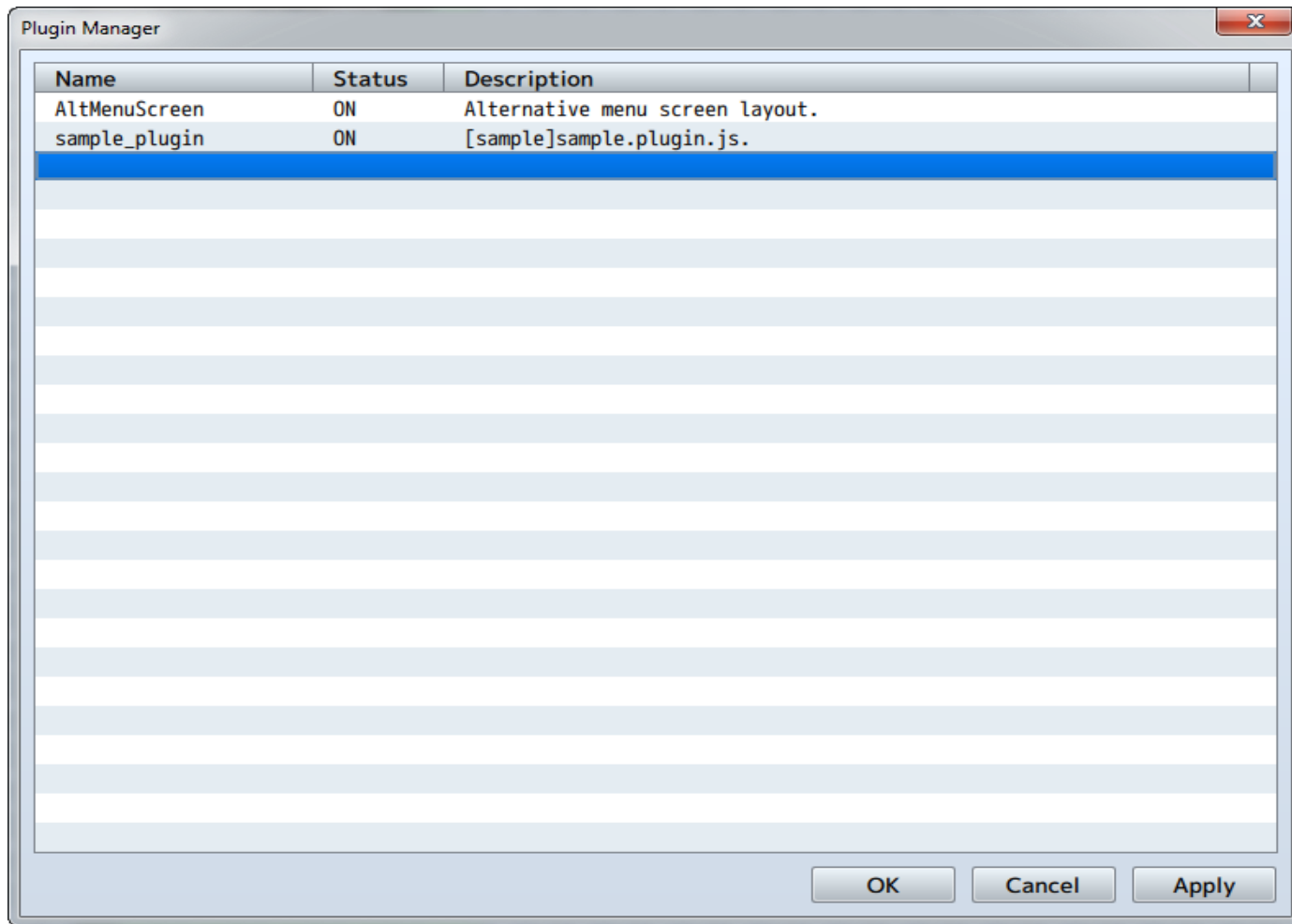
Changing multiple [Statuses] at once

Press the Shift key to select multiple plugins and right-click to change all statuses at once.

Plugin Manager		
Name	**Status**	**Description**
AltMenuScreen	ON	Alternative menu screen layout.
AltSaveScreen	ON	Alternative save/load screen layout.
EnemyBook	OFF	Displays detailed statuses of enemies.
ItemBook	OFF	Displays detailed statuses of items.

Edit...	Return
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Turn ON	
Turn OFF	
Delete	Del
Select All	Ctrl+A

OK   Cancel   Apply
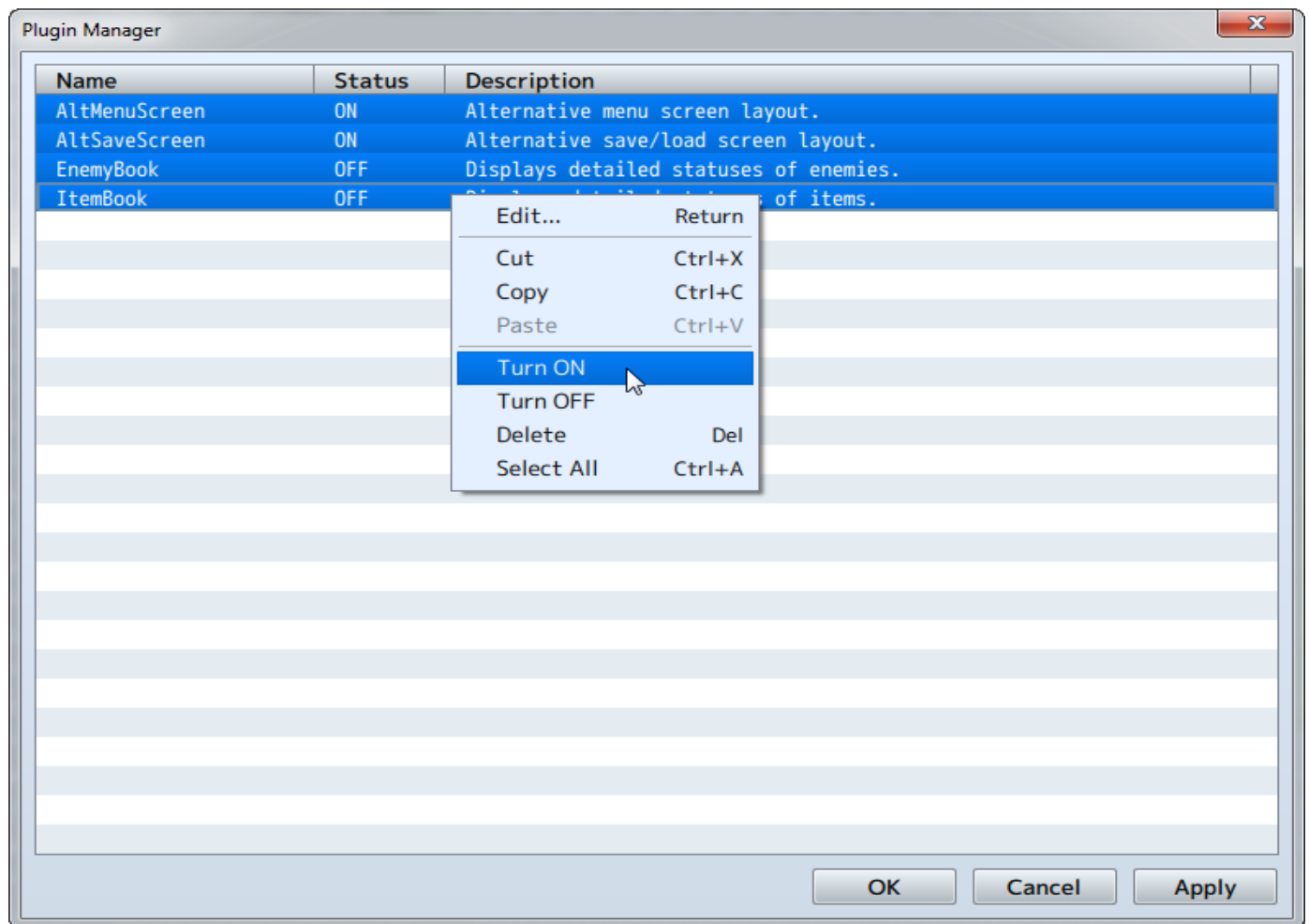
Using Official Plugins

In RPG Maker MV, there are several plugins already included in your new project and can be used with simple settings.

AltMenuScreen

A plugin that changes the Menu Screen to a different layout.
Party members will be displayed horizontally.
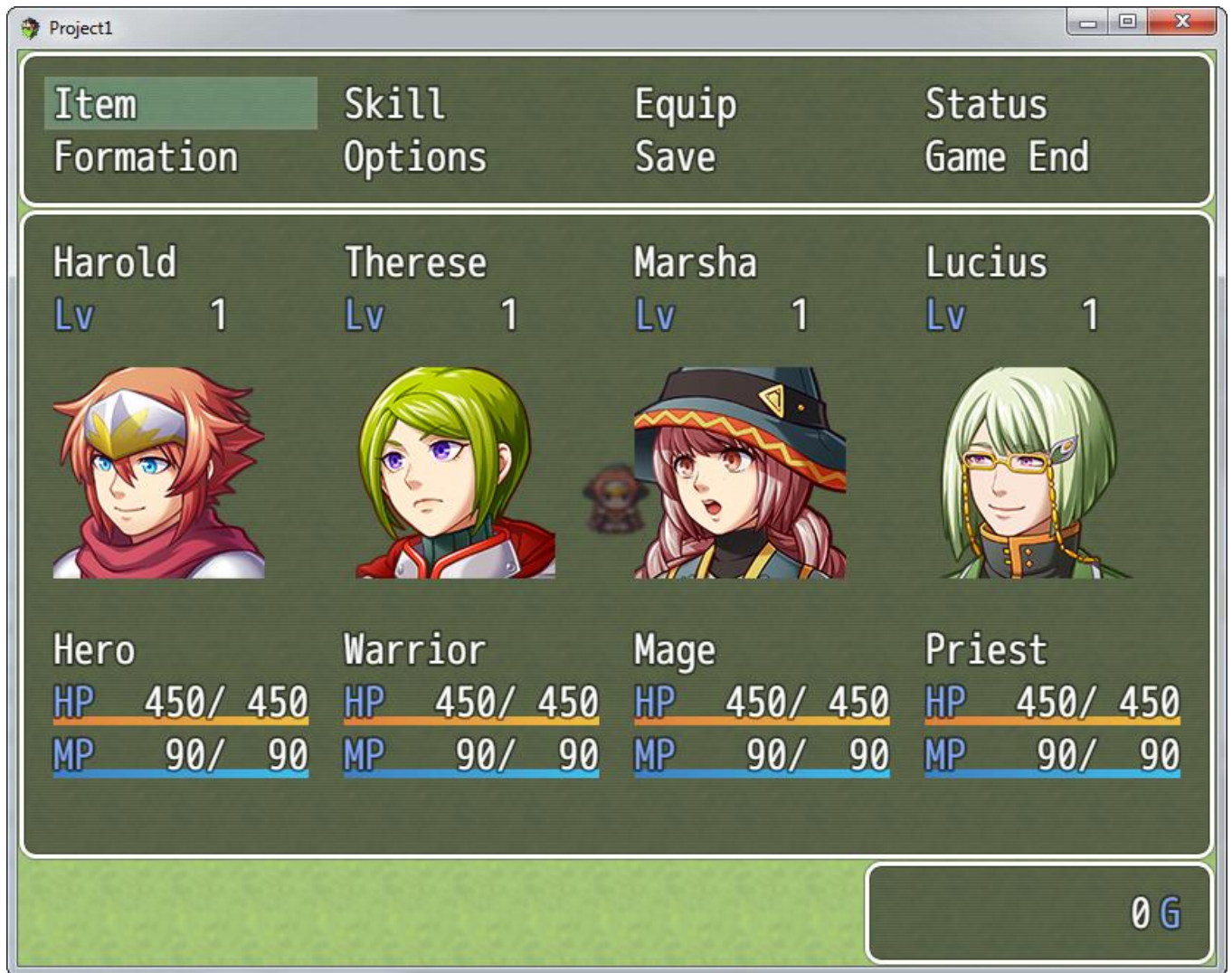
AltSaveScreen

A plugin that changes the Save Screen and Load Screen to a different layout.
It becomes possible to display images such as the faces of party members.
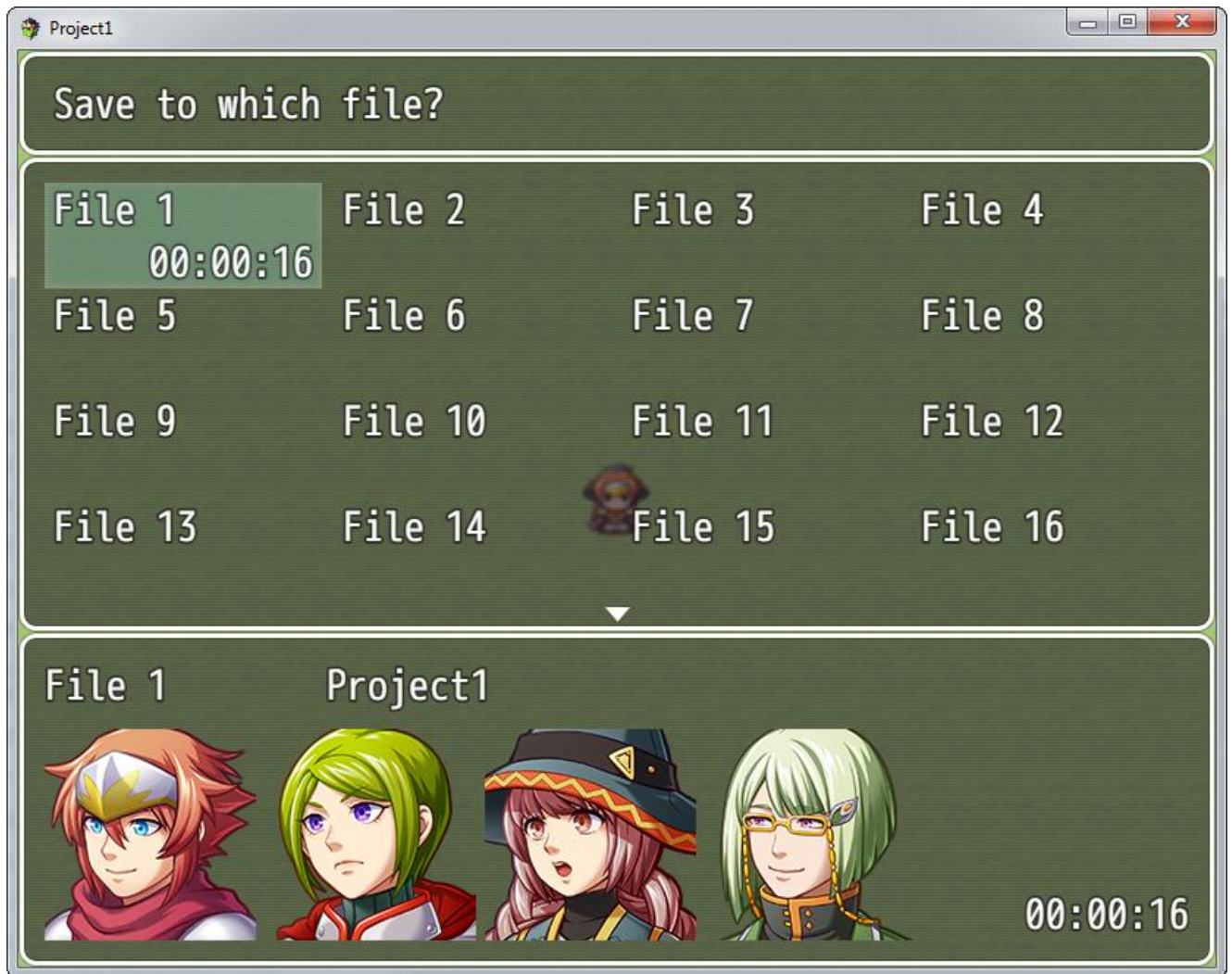
EnemyBook

Monster Enemy Book
You can make a [Monster Enemy Book] which lists the enemies the player has fought.

Plugin Command
Set up as an event using the [Plugin Command] advanced event command.
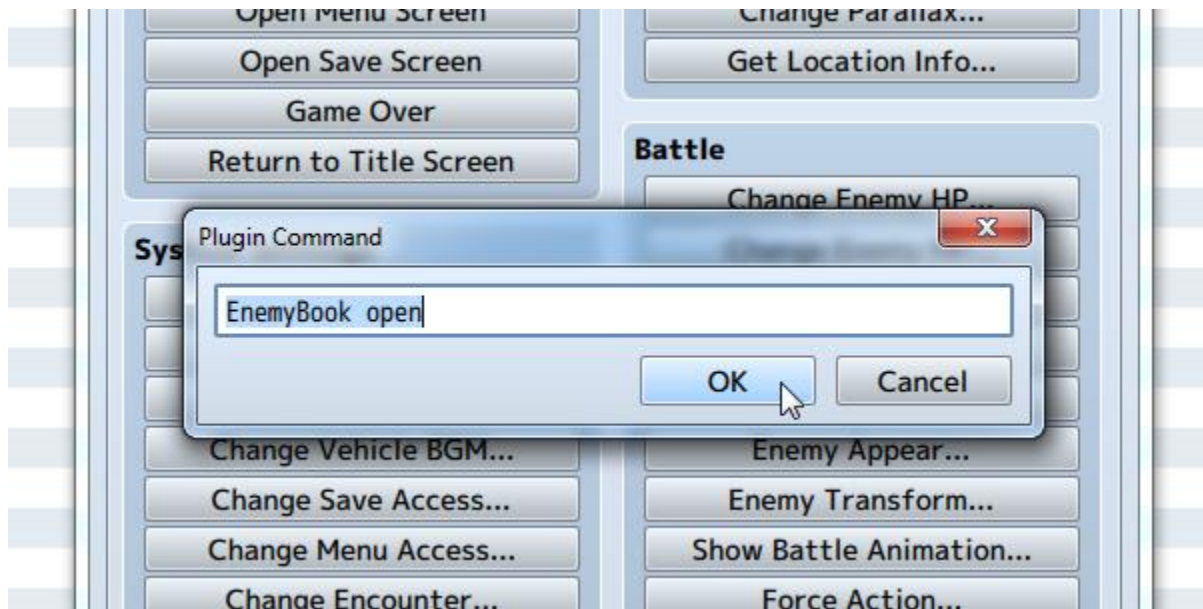
EnemyBook open	Open the Enemy Book Screen
EnemyBook add 3	Add the enemy 3 to the Enemy Book
EnemyBook remove 4	Remove enemy 4 from the Enemy Book
EnemyBook complete	Complete the Enemy Book
EnemyBook clear	Clear the Enemy Book

Enemy Note

Writes to the [Note] section in the [Enemies] tab of the database.

General Settings

Name: Bat

Max HP: 200

Max MP: 0

Image:

Attack: 30

Defense: 30

M.Attack: 30

M.Defense: 30

Agility: 30

Luck: 30

Traits

Type	Content	
Ex-Parameter	Hit Rate + 95%	
Ex-Parameter	Evasion Rate + 5%	
Attack Element	Physical	

Rewards

EXP: 0

Gold: 0

Drop Items

None	...
None	...
None	...

Action Patterns

Skill	Condition	R
Attack	Always	5

Note

```
<desc1:description 1>
<desc2:description 2>
```

<desc1:description 1>	Display description on the first line
<desc2:description 2>	Display description on the second line
<book:no>	Use when you will not put in the Enemy Book

ItemBook

Item Book
You can make an [Item Book] which lists all the equipment and items a player has collected.

Plugin Command
Set up as an event using the [Plugin Command] advanced event command.
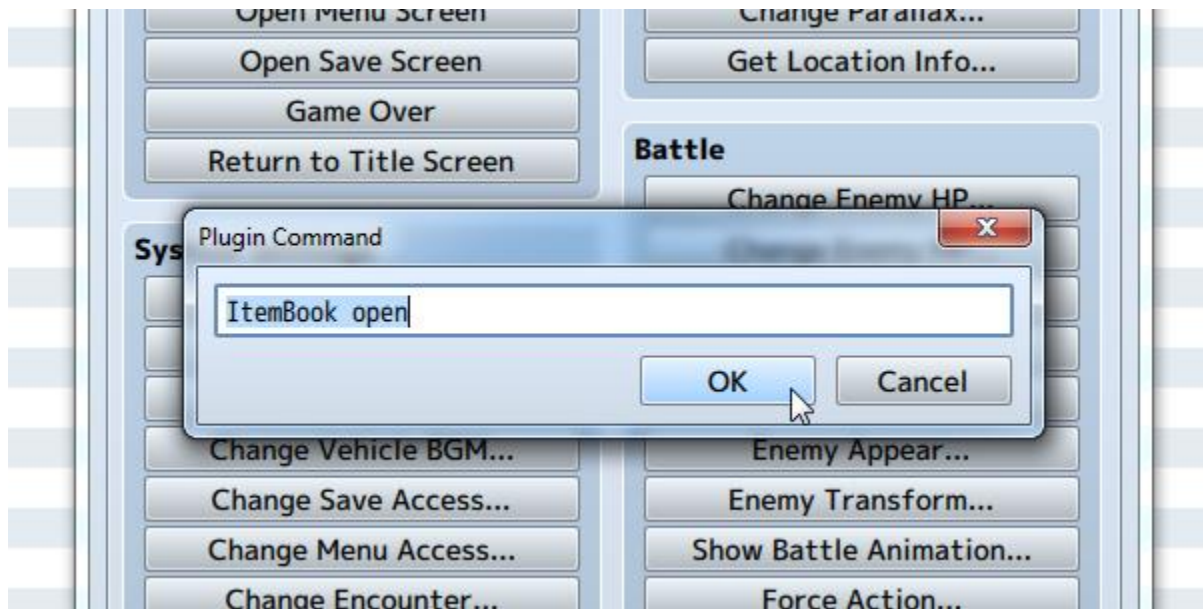
ItemBook open	Open the Item Book Screen
ItemBook add weapon 3	Add weapon 3 to the Item Book
ItemBook add armor 4	Add armor 4 to the Item Book
ItemBook remove armor 5	Remove armor 5 from the Item Book
ItemBook remove item 6	Remove item 6 from the Item Book
ItemBook complete	Complete the Item Book
ItemBook clear	Clear the Item Book

Item (weapons, armors) Note

Writes to the [Note] field in the [Items (weapons, armors)] tab in the database.

General Settings

Name:

Potion

Icon:

176

Description:

Item Type:

Regular Item

Price:

50

Consumable:

Yes

Scope:

1 Ally

Occasion:

Always

Invocation

Speed:

0

Success:

100 %

Repeat:

1

TP Gain:

0

Hit Type:

Certain Hit

Animation:

Heal One 1

Damage

Type:

None

Element:

Formula:

Variance:

Critical Hits:

Effects

Type	Content	
Recover HP	500	

Note

<book:no>

<book:no>	Use when you will not put in the Item Book

SimpleMsgSideView

Hide the battle log shown in side-view during battle. Only the names of skills, items and actions (Attack and Guard) will be displayed in the upper part of the screen.

Parameters

displayAttack

Whether to display a normal [Attack] or not.
Will not be displayed when set to 0 (default), and will be displayed when set to 1.

position

Whether to display the names of skills and items to the left or in the center.
Left when set to 0, and in the center when set to 1.

TitleCommandPosition

A plugin that changes the position of the command window on the Title Screen and its background.

Parameters
  Offset X

   The offset value for the X coordinate.

  Offset Y

   The offset value for the Y coordinate.

  Width

   Width of the command window.

  Background

   The background type. You can select from 3 different types: 0 - Normal, 1 - Dark, 2 - Transparent.

WeaponSkill

A plugin that changes the skill ID for the normal attacks for each weapon.

Create the skill that you want to add to the weapon. Specifying the skill ID in the [Note] field of the weapon will cause that skill to be activated when attacking with a weapon.
Example: <skill_id:3>

You can create the following types of weapons.

- Make a weapon that attacks multiple enemies.

- Make a weapon that attacks 2 or 3 times.

- If recovery magic is set to a skill, you can select an ally and cause them to recover when you select [Attack].

- It is possible to create weapons which are equal to the Guard command, etc.

Set the [Skill Type] to "None" for skills selected using weapons.
In the case that a Skill Type whose use can be disabled, the player will not be able to select "Attack" if that skill is disabled.

Plugin Specifications

In RPG Maker MV, you can use JavaScript to create your own plugins. Below is information for developers regarding creating plugins.
The following information is for those developers who want to create original plugins using JavaScript.

Basics

- A plugin's .js file will be placed in the js/plugins folder.

- The editor will write to the js/plugins.js file, storing the name of the plugins that will be used and their parameters.

- In order to limit the scope of variables in plugin scripts, they will all be enclosed in immediate functions.

- UTF-8 will be used for the character code.

Redefining Methods

- Redefine the methods which you want to change the behaviors of after saving to a local variable as necessary.

- To minimize conflicts between plugins, it is best to add behaviors that are as original as possible.

Parameters

- Descriptions for plugins and their parameters displayed in the editor are specified in comments that start with "/*:".

@plugindesc	The plugin's description.
@author	The plugin's author.
@param	Name of the parameters.
@desc	Description of the parameters.
@default	Default values of the parameters.
@help	A detailed description of the plugin.

- Use PluginManager.parameters() to get the parameters specified by the editor.

- All parameters will be treated as strings so convert them as necessary.

Metadata

The [Note] field found in each item of the database can be used to define unique data used with each plugin.

```
<name:data>
```

In this way, data which has been written in a fixed format will be automatically developed inside a "meta" variable by standard scripts.
In the case above, the following conditions will be met (objects will be treated as data).

```
object.meta.name === 'data'
```

Plugin Command

Plugin commands are used for easily defining unique event processes for plugins. When implementing these, the pluginCommand of the Game_Interpreter class will be redefined in the following way.

```
var _Game_Interpreter_pluginCommand =
        Game_Interpreter.prototype.pluginCommand;
Game_Interpreter.prototype.pluginCommand = function(command, args) {
    _Game_Interpreter_pluginCommand.call(this, command, args);
    // insert additional processing details here
};
```

The contents of the plugin commands called by the user will pass through the function command and args methods. Commands will be strings, and args will be an array of strings. For example, when evaluating whether or not the user has entered in "MyPlugin clear", the following will occur.

```
if (command === 'MyPlugin' && args[0] === 'clear') {
```

```
            }
```

Just like the parameters for a plugin, everything will be passed as a string to convert them as necessary.

Multi-language Support

You can specify the language code following the "/*:" in the beginning of the comment block. This will become "/*:ja" when setting this to Japanese. Comment blocks with a language code specified will only be used in that language's editor environment, the unlabeled part (normally English) will be used when a language is not supported.

Codes	Language
ja	Japanese
fr	French
de	German
es	Spanish
it	Italian
pt	Portuguese
ru	Russian
zh	Chinese
ko	Korean

Output Formats

Deployment

Exports the deployment package for each platform. It is possible to output to Windows, Mac OS X, Android and iOS.

Windows

The exported game folder will be deployed.
Please read [Using Enigma Virtual Box] when wanting to package the game folder and distribute only the .exe file.

Mac OS X

Distributes the exported game folder or the Game.app file inside the game folder.
* Game.app will be displayed as a folder in a Windows environment, but you can run this as an application in a Mac OS X environment.

Running Your Game On Android/iOS

Please view [Converting to an Android App] and [Converting to an iOS App].

When Publishing on the Web

You can play the game in the browser by simply uploading the project folder onto a Web server.
* The project folder will be located in the location specified in "Location" when creating a new project and saving.
* The Game.rpgproject and save folder are not needed to run the game, but there will not be any issues even if you upload them.

Gameplay System Requirements

System Requirements	Environments which support HTML5 Canvas and Web Audio API iOS 8 or higher Android 4.4 or higher
Recommended Environments	Windows, Android: Google Chrome Mac OS X, iOS: Safari
* Please use Wi-Fi when playing using a browser on a smartphone. * A browser that supports WebGL is recommended for gameplay.	

Converting to an Android App <Creating an apk file>

1 Overview

From this version, games created using RPG Maker MV will now be played using browsers which support HTML5 technology such as WebGL & 2DCanvas, WebAudio and JavaScript. There is also continued support for HTML5 in recent browsers for Android, allowing users to enjoy playing games made using RPG Maker.

Rather than explaining how to play using a browser, we will explain in this section how to play your game as an Android application. Running an HTML5 game as an application using an Android device is similar to running a game as an Android application using a browser that supports HTML5. This method of development of running a web application as a native application is known by such names as "Hybrid App", and is popular from the fact that it can be run on cross-platforms.

Browsers which package games as apps are "Browser Engines" which handle the Web display and sounds, and have Android built-in (called WebView). However, browser engines with Android built-in seem to be lagging behind a

little when it comes to supporting HTML5. Because of this reason, we will be giving instructions which use a technology called "Crosswalk Project" that uses the Chrome browser which has a high level of support for HTML5. A similar explanation to the explanation found on this page can be viewed on the Crosswalk Project website so please check it out. ([Crosswalk Project on Android]).

Operating Systems

The work environments used for this explanation will be Windows 8.1 and OS X 10.10. Additionally, the computer used to do your work will be referred to as the "Host PC".

2 Prepare the Packaging Environment

Installing games created using RPG Maker on Android will require creating an apk file. An apk file is the installer for Android apps and its extension is apk. This is compressed in a ZIP format, and contains various necessary files for running the application. We will refer to creating the apk file here as "Packaging".

Also, an Internet connection is required to download some of the necessary developer tools.

2-1 Python Installation and Setup

A packaging script is provided in Crosswalk Project so that you can easily package your game. This is made using a scripting language called "Python" so we will set up an environment where Python can be run. This section explains how to install Python on Windows but **Python is already installed with OS X so there is no need to install it. Please proceed to 2-2.**

2-1-1 Downloading & Installing Python

Start downloading the installer which can be found on https://www.python.org/downloads/. It is not particularly necessary to change any of the items during installation if they are not necessary, but it is a good idea to check where you are installing Python.

Python has been installed to C:\Python34.
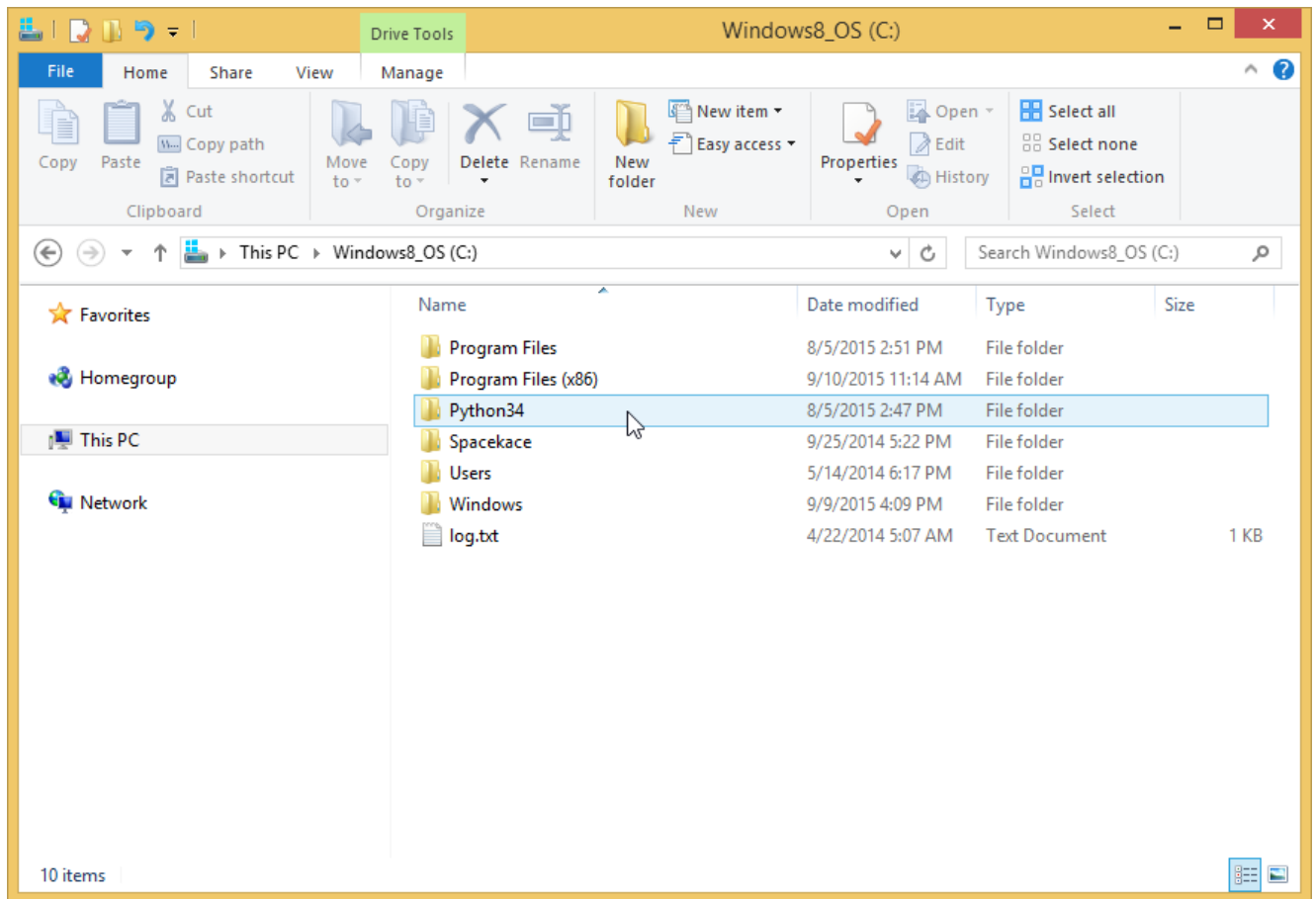
2-1-2 Path Setup & Check

The path will be set to C:\Python34 where Python was installed so that Python commands will be easier to use.
Right-click "This PC" and open up [Advanced system settings] from [Properties].
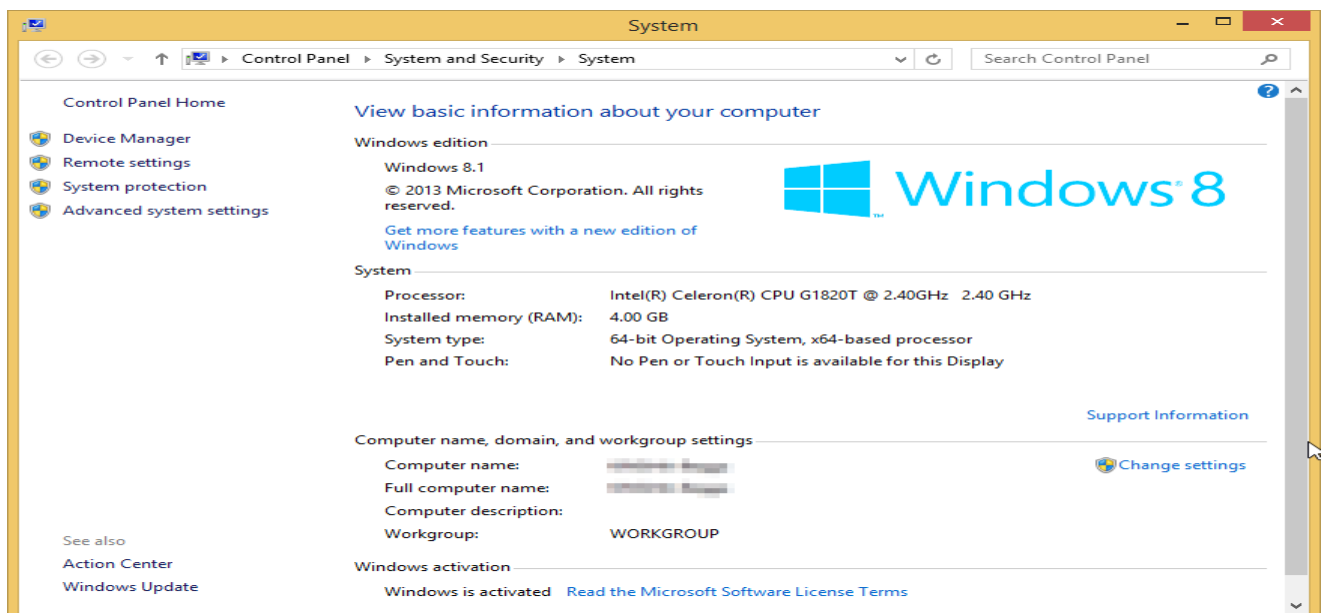
Click [Environment Variables] in [System Properties]. Select [Path] under [System Variables] and click the Edit button.
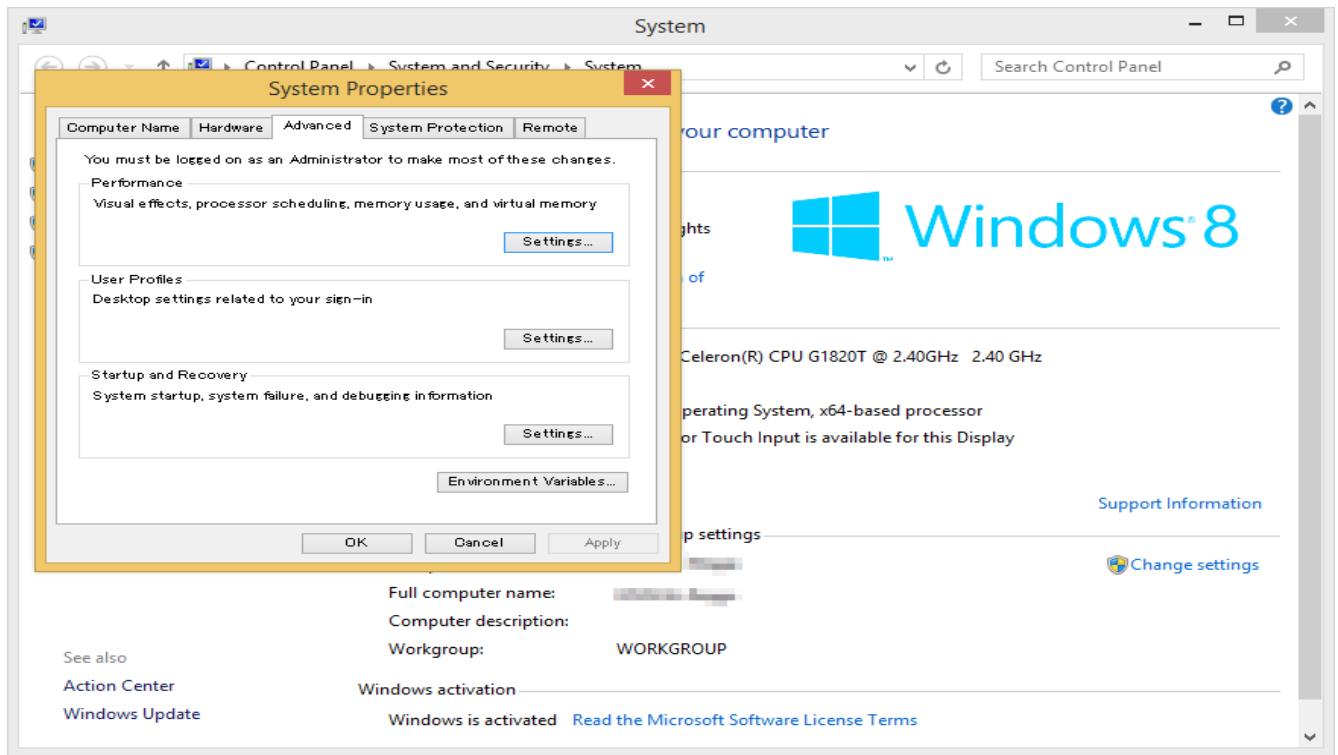
Edit the variables. Adding ";C:\Python34\" to the end of the original string will allow the path to be used (the semicolon is a break character).

```
;C:\Python34\
```
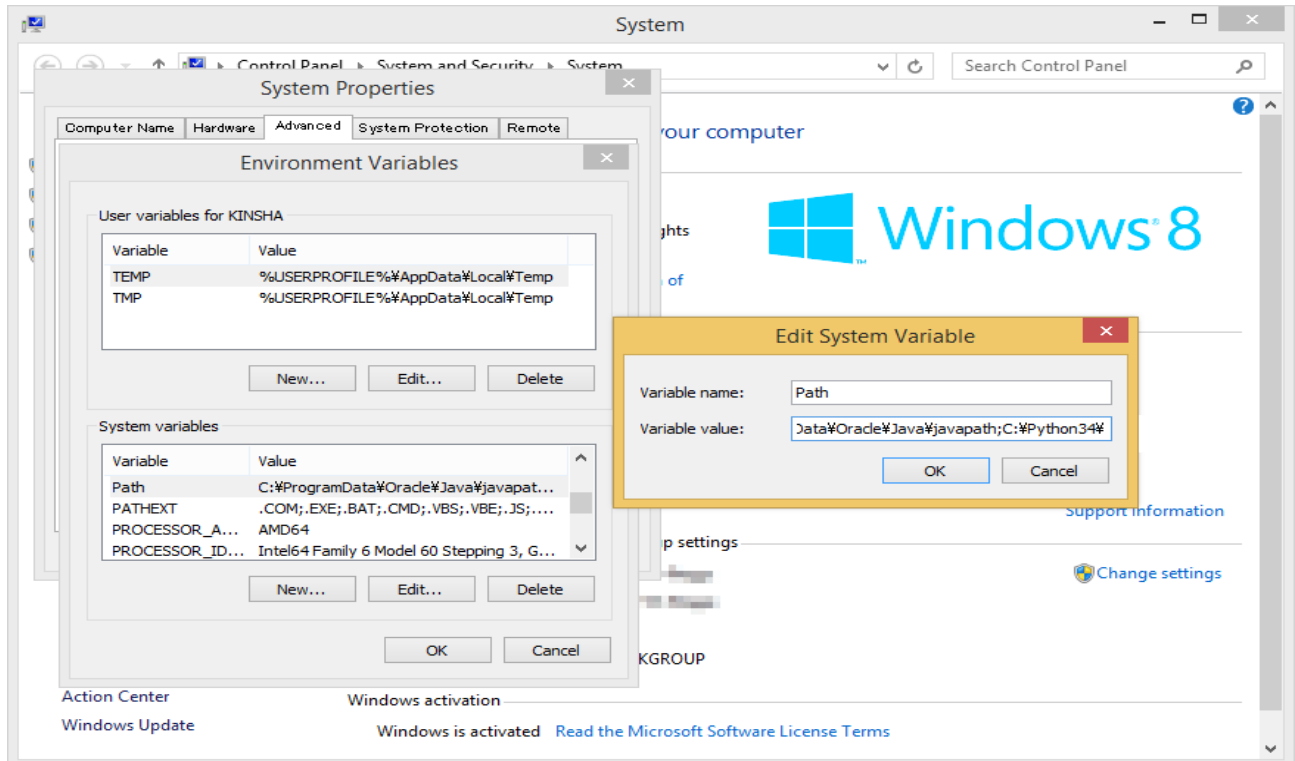

Confirm using command prompt. When using Windows 8.1, open the C drive in the explorer, and select [Open command prompt].
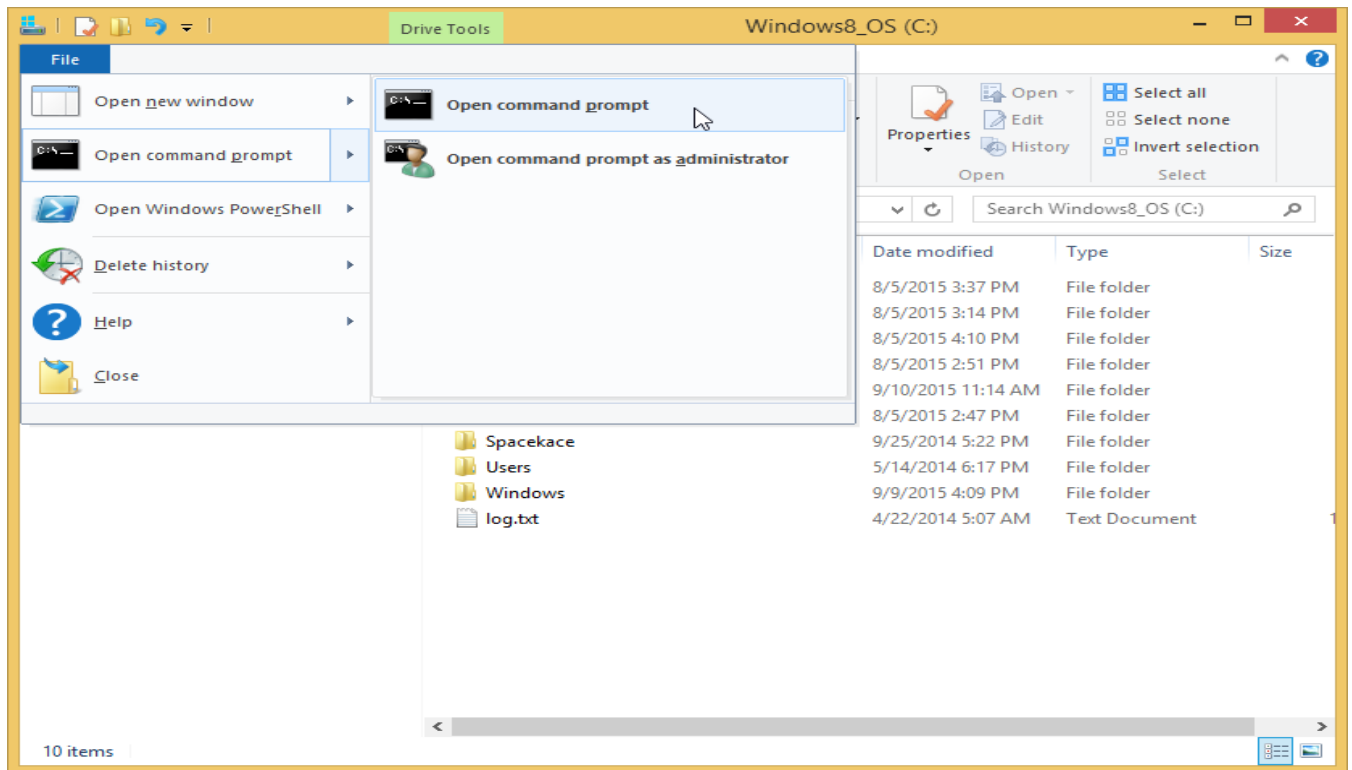
Enter "python --version" command and press the Enter key. If the version of Python is displayed, that means there is no problem.

```
> python --version
```
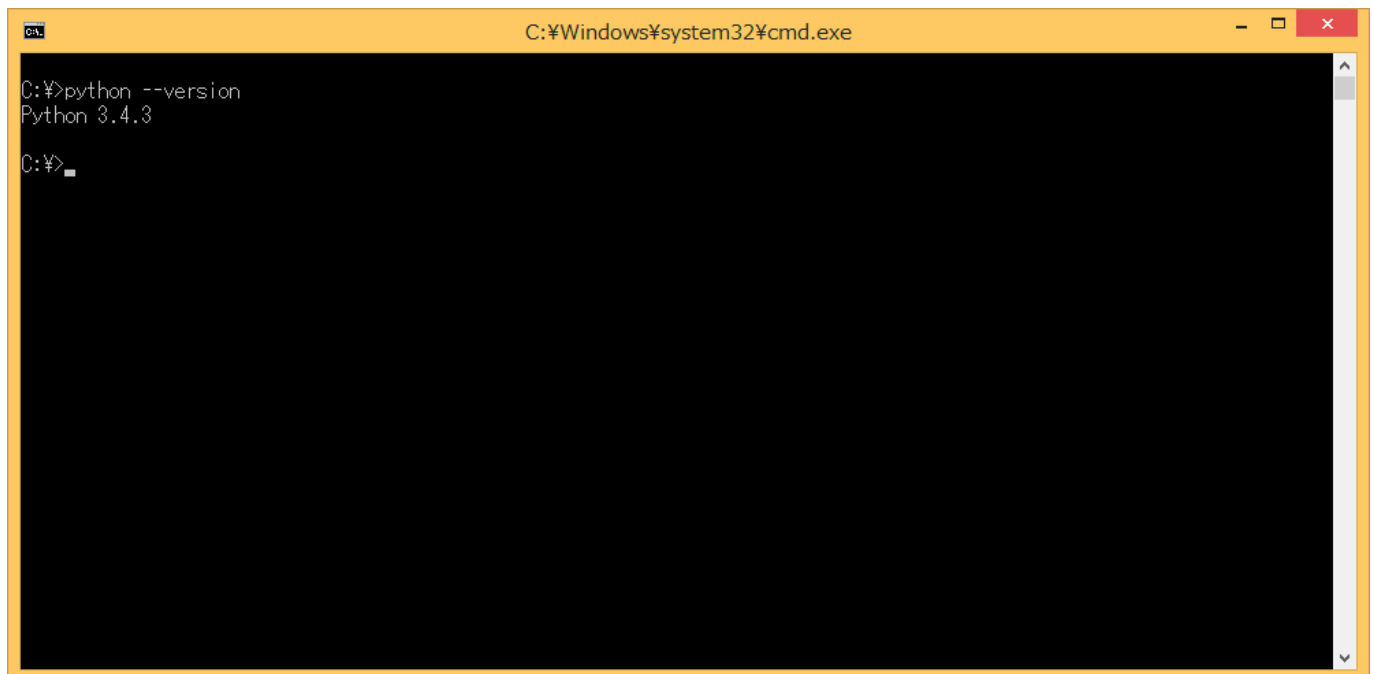

2-2 Installing & Setting Up Oracle JDK

In order to run Android SDK tools and a tool known as "Ant," you will need JDK (Java Development Kit). This is current provided by Oracle. Since JDK has stopped being included in OS X from 10.6 and on, you will need to install it.

2-2-1 Downloading & Installing JDK

Download JDK from http://www.oracle.com/technetwork/java/javase/downloads/index.html. The newest version of JDK is Java SE 8u51 as of August 2015.

Download the version which is appropriate for your system. There are two versions for Windows, 32-bit (Windows x86) and 64-bit (Windows x64), so be sure to choose the correct one.
Select "Accept License Agreement" using the above radio button.

Run the downloaded installer to install Java SE Development Kit. You do not need to change any of the options.
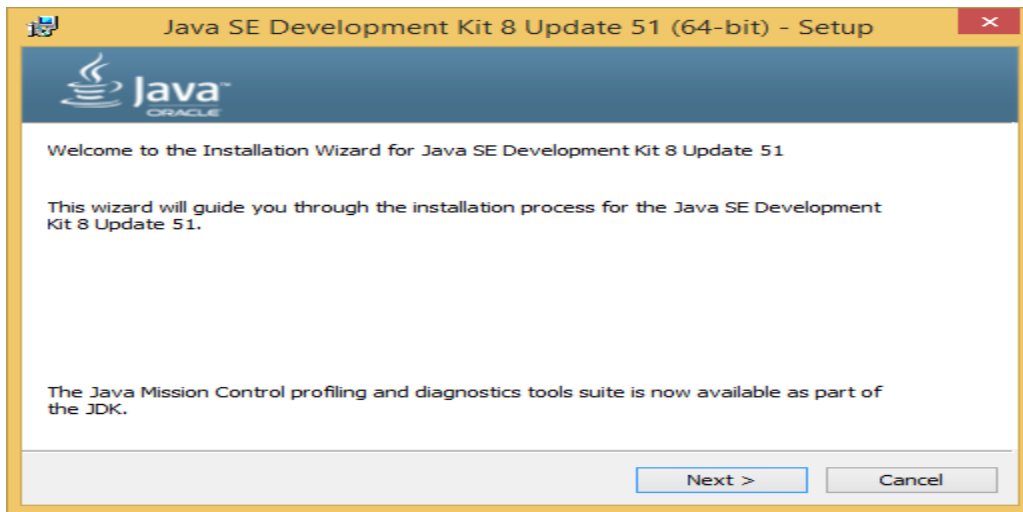
2-2-2 Configuring and Checking Environment Variables & Path

We will set up the JAVA_HOME Environment Variable. Just as you set up the path for Python in 2-1-2, open [System Properties] and click the [New] button in [Environment Variables].
Enter "JAVA_HOME" for the variable name and input the installation location for JDK for the variable value. The variable here was set to C:\Program Files\Java\jdk1.8.0_51.

```
JAVA_HOME

C:\Program Files\Java\jdk1.8.0_51
```
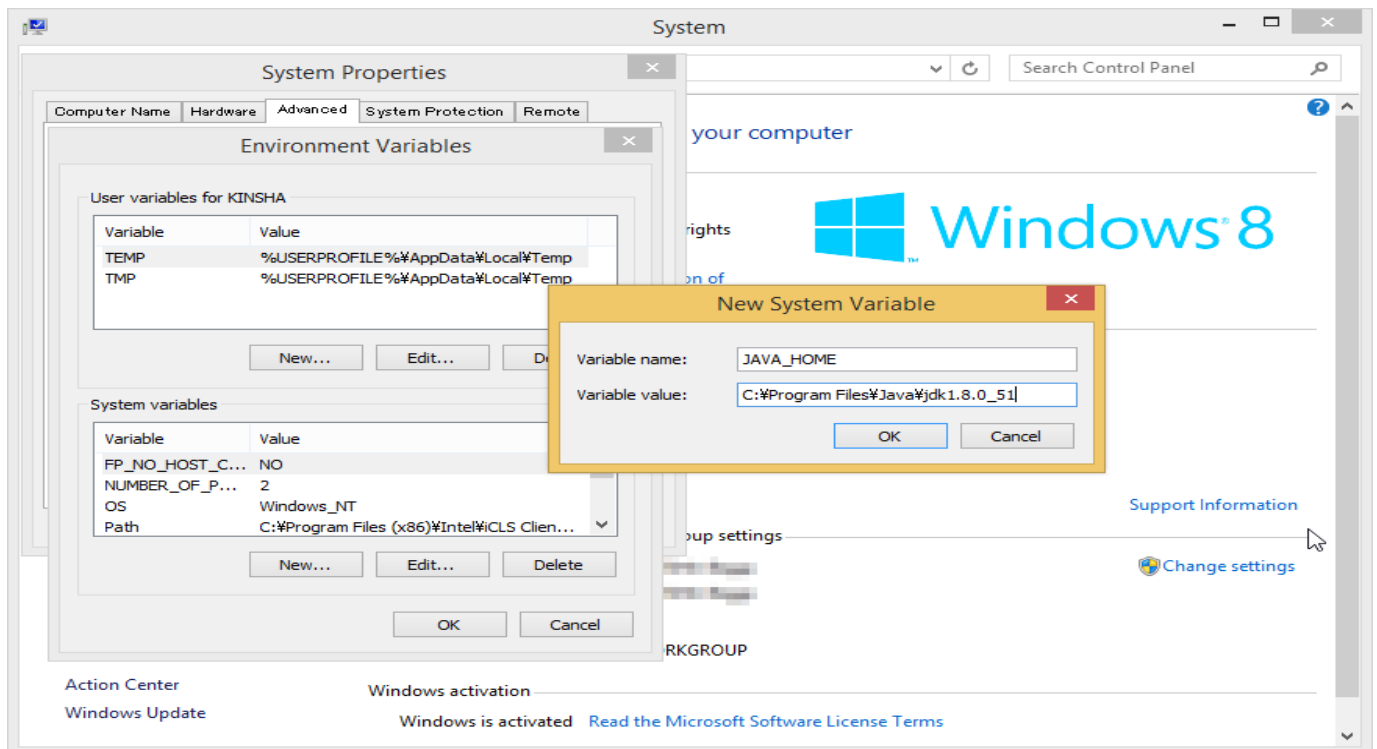

This creates the path. Select [Path] under [System Variables] and click the Edit button. Enter ;%JAVA_HOME%\bin\ following the value already entered.

```
;%JAVA_HOME%\bin\
```
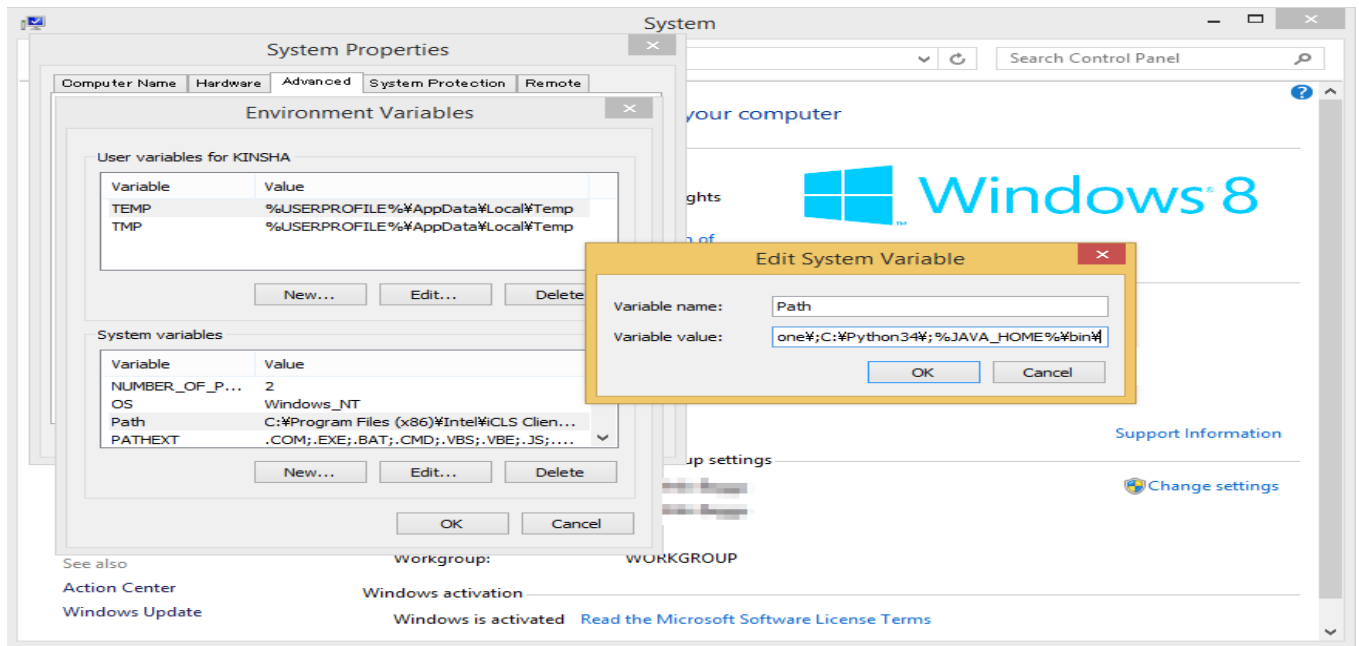
Check the version using Command Prompt just as you did with Python.
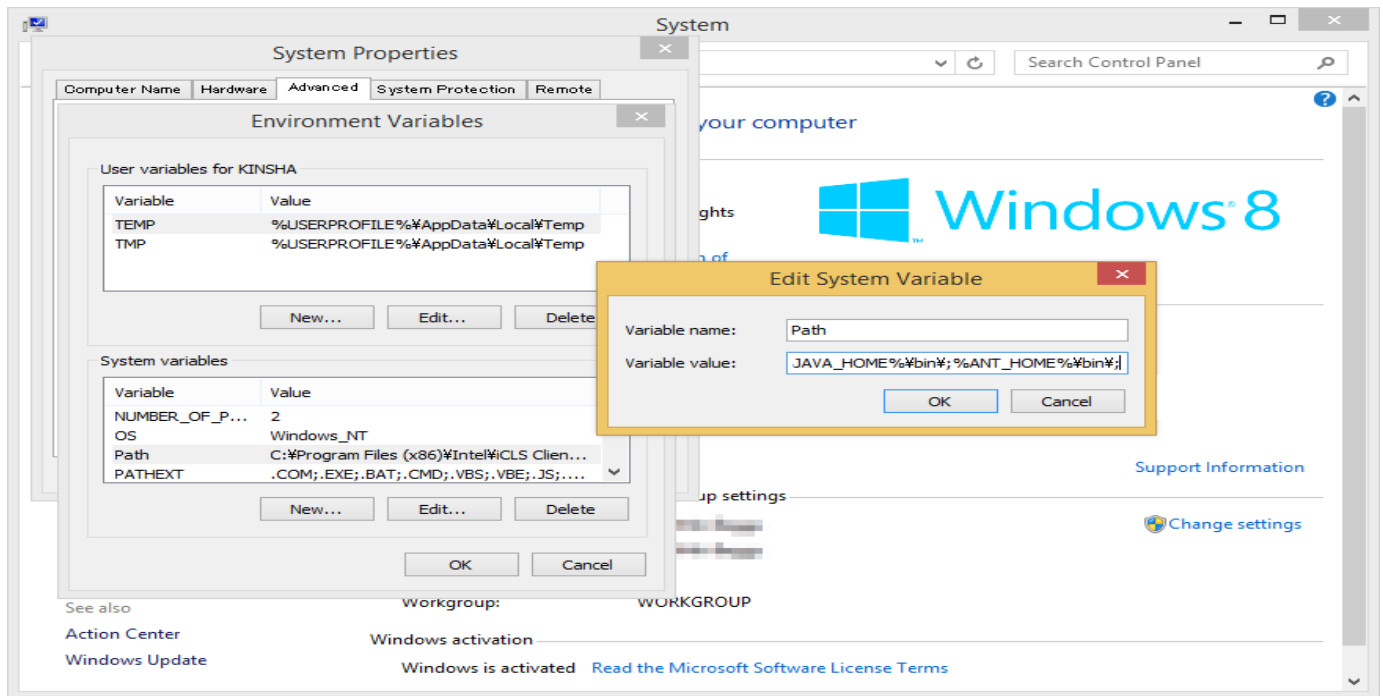Enter "java --version" command and press the Enter key. If the version is displayed, that means there is no problem.

```
> java -version
```
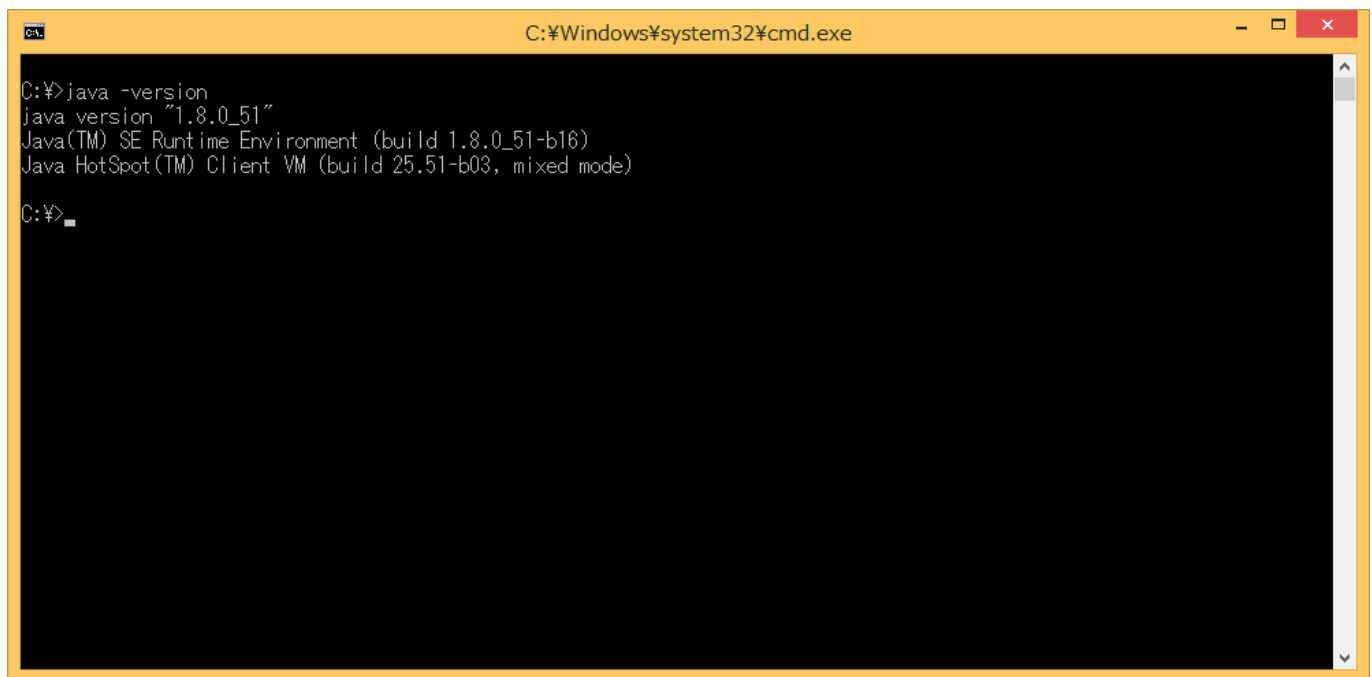

2-3 Installing and Configuring Apache Ant

Ant is known as a build tool that is run using Java. It is utilized internally using a packaging script. You need to install Ant on both Windows and OS X.

2-3-1 Downloading & Installing Ant

Download the Ant binaries from http://ant.apache.org/bindownload.cgi.
Download the [.zip archive:] found under the [Current Release of Ant] section. The newest version of Ant is 1.9.6 as of August 2015.

ApacheCon
™
Europe 2015
Sep 28 - Oct 2
Budapest Hungary

Ant is distributed as `zip`, `tar.gz` and `tar.bz2` archives - the contents are the same. Please note that the `tar.*` archives contain file names longer than 100 characters and have been created using GNU tar extensions. Thus they must be untarred with a GNU compatible version of `tar`.

In addition the JPackage project provides RPMs at their own distribution site.

If you do not see the file you need in the links below, please see the master distribution directory or, preferably, its mirror.

Mirror

You are currently using **http://ftp.tsukuba.wide.ad.jp/software/apache/**. If you encounter a problem with this mirror, please select another mirror. If all mirrors are failing, there are *backup* mirrors (at the end of the mirrors list) that should be available.

Other mirrors: [http://ftp.tsukuba.wide.ad.jp/software/apache/ ▾] [Change]

Current Release of Ant

Currently, Apache Ant 1.9.6 is the best available version, see the release notes.

Note
Ant 1.9.6 was released on 02-Jul-2015 and may not be available on all mirrors for a few days.

Tar files may require gnu tar to extract
Tar files in the distribution contain long file names, and may require gnu tar to do the extraction.

- `.zip` archive: apache-ant-1.9.6-bin.zip [PGP] [SHA1] [SHA512] [MD5]
- `.tar.gz` archive: apache-ant-1.9.6-bin.tar.gz [PGP] [SHA1] [SHA512] [MD5]
- `.tar.bz2` archive: apache-ant-1.9.6-bin.tar.bz2 [PGP] [SHA1] [SHA512] [MD5]

Old Ant Releases

Older releases of Ant can be found here. We highly recommend to not use those releases but upgrade to Ant's latest release.

Verify Releases

The file you downloaded is a compressed ZIP file so extract it in a location of your choosing. In this example, Ant was extracted directly to the C drive.
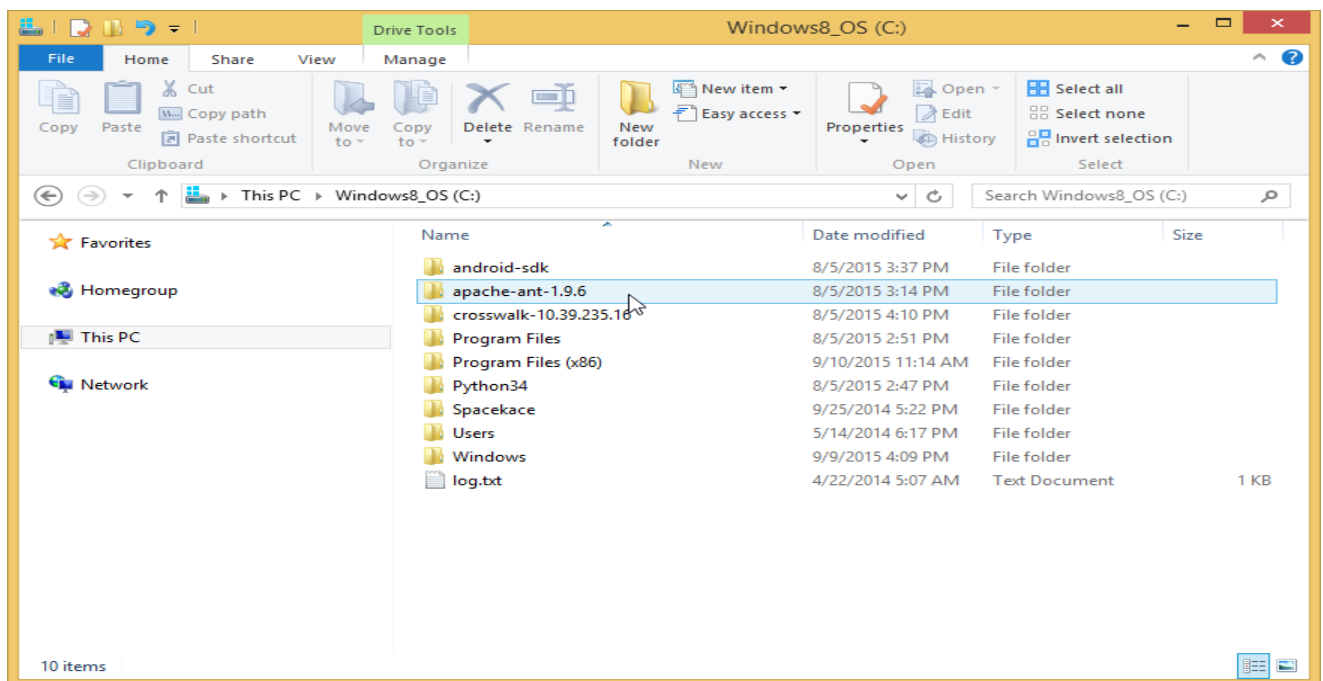
2-3-2 Configuring and Checking Environment Variables & Path
Here we will set up the path and environment variables for Ant. Just as when we set up the path to Python in 2-1-2, open the Environment Variables window.

First, we will configure the ANT_HOME environment variable. Click the New button under System Environment Variables, and enter [ANT_HOME] as the variable name and the path where Ant was extracted as the variable value.

```
ANT_HOME

C:\apache-ant-1.9.6
```
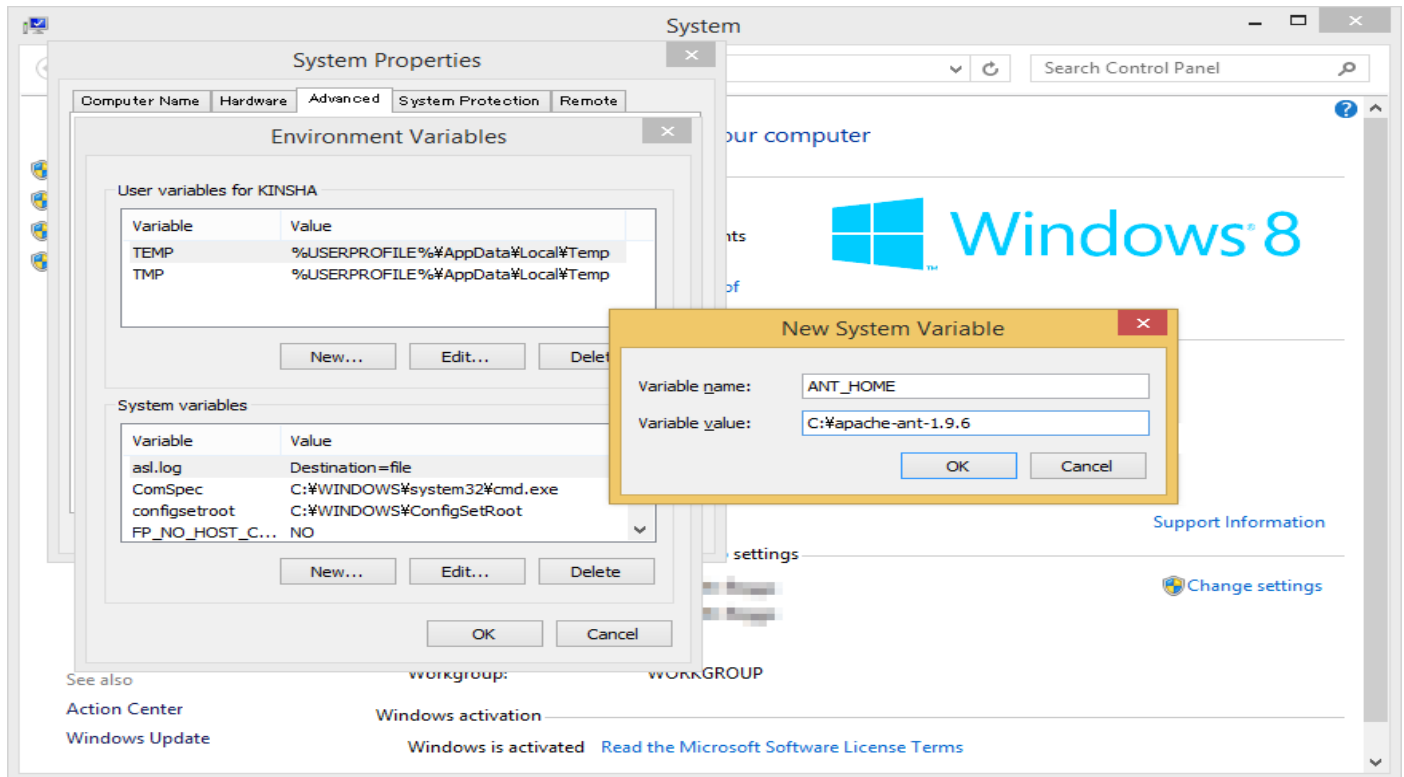

This creates the path. Select [Path] under [System Variables] and click the Edit button. Enter ";%ANT_HOME%\bin\" following the value already entered.

```
;%ANT_HOME%\bin\
```

To check to make sure that the path is set up, use Command Prompt to check the version just as you have up until now. Enter "ant --version" command and press the Enter key.

```
> ant -version
```
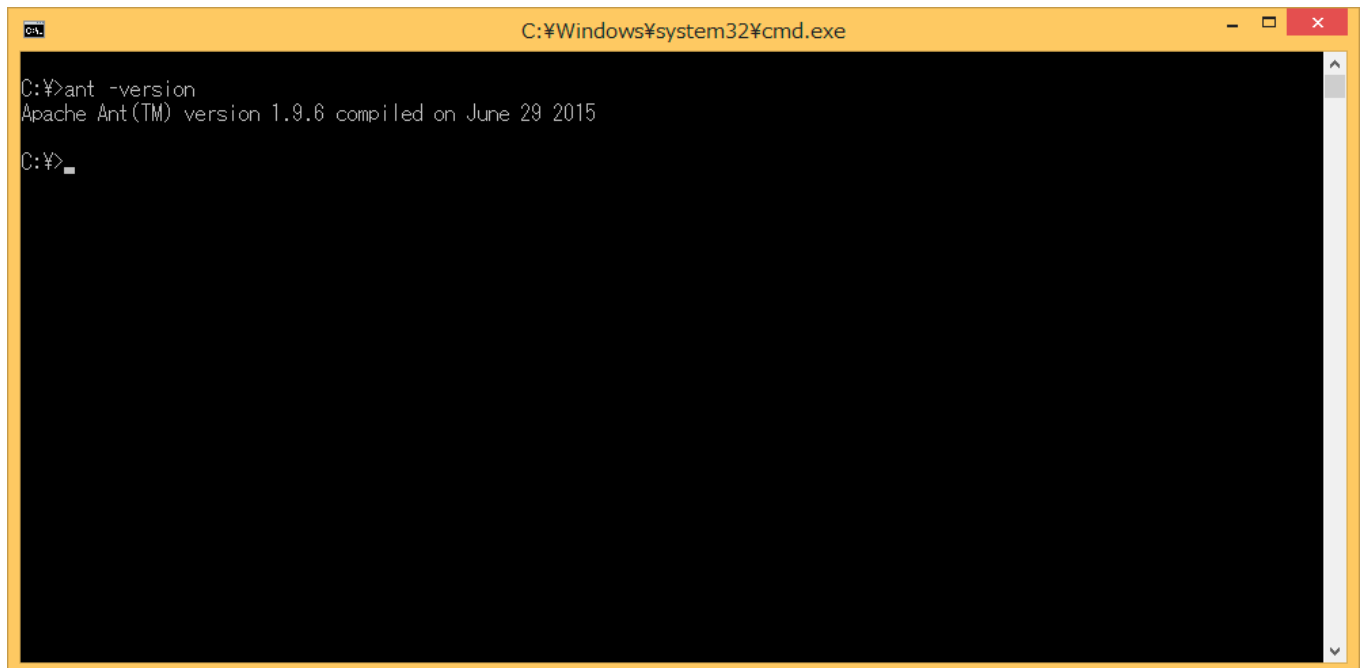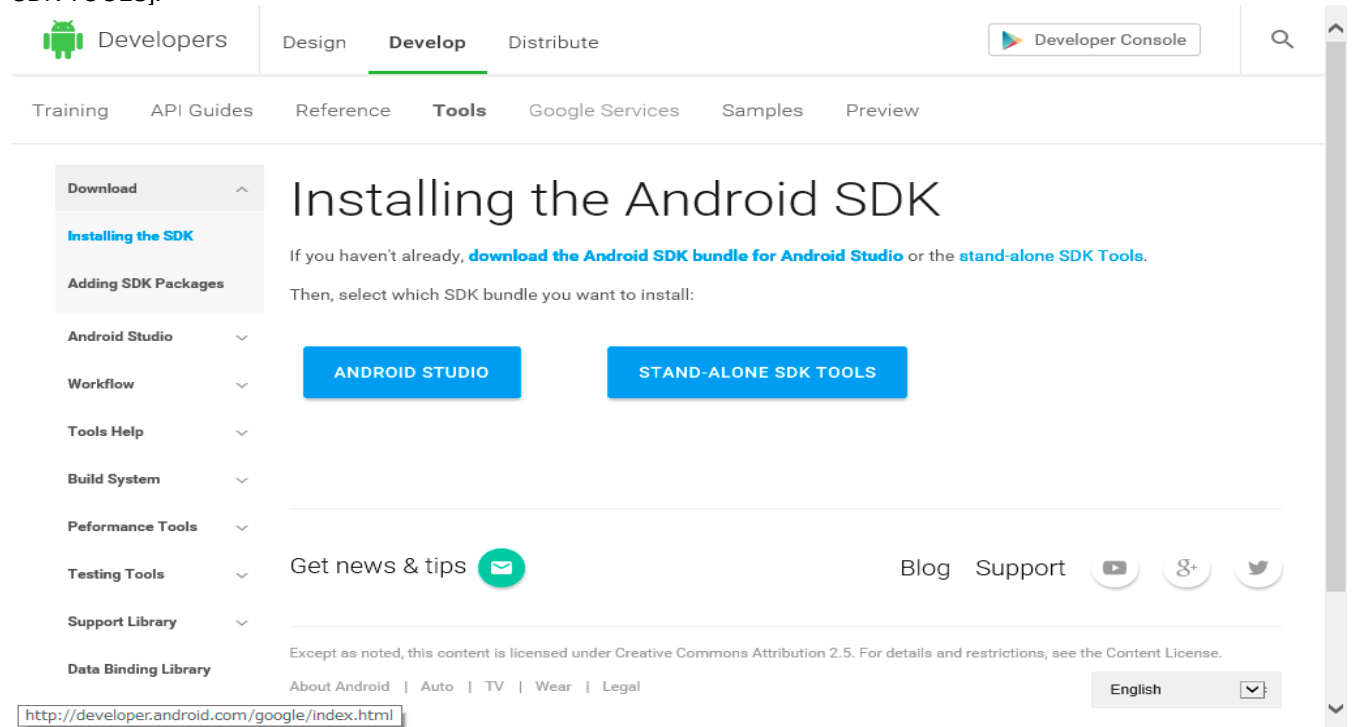

2-4 Installing and Configuring Android SDK

SDK is short for "Software Development Kit" and just as the name says, this contains a kit for developing software. Necessary files for packaging and convenient tools for testing are included in the kit.

2-4-1 Downloading & Installing Android SDK

Download the Android SDK from http://developer.android.com/sdk/installing/index.html. In the SDK, you can find [ANDROID STUDIO], software for developing Android apps. However, here we will be using the [STAND-ALONE SDK TOOLS].
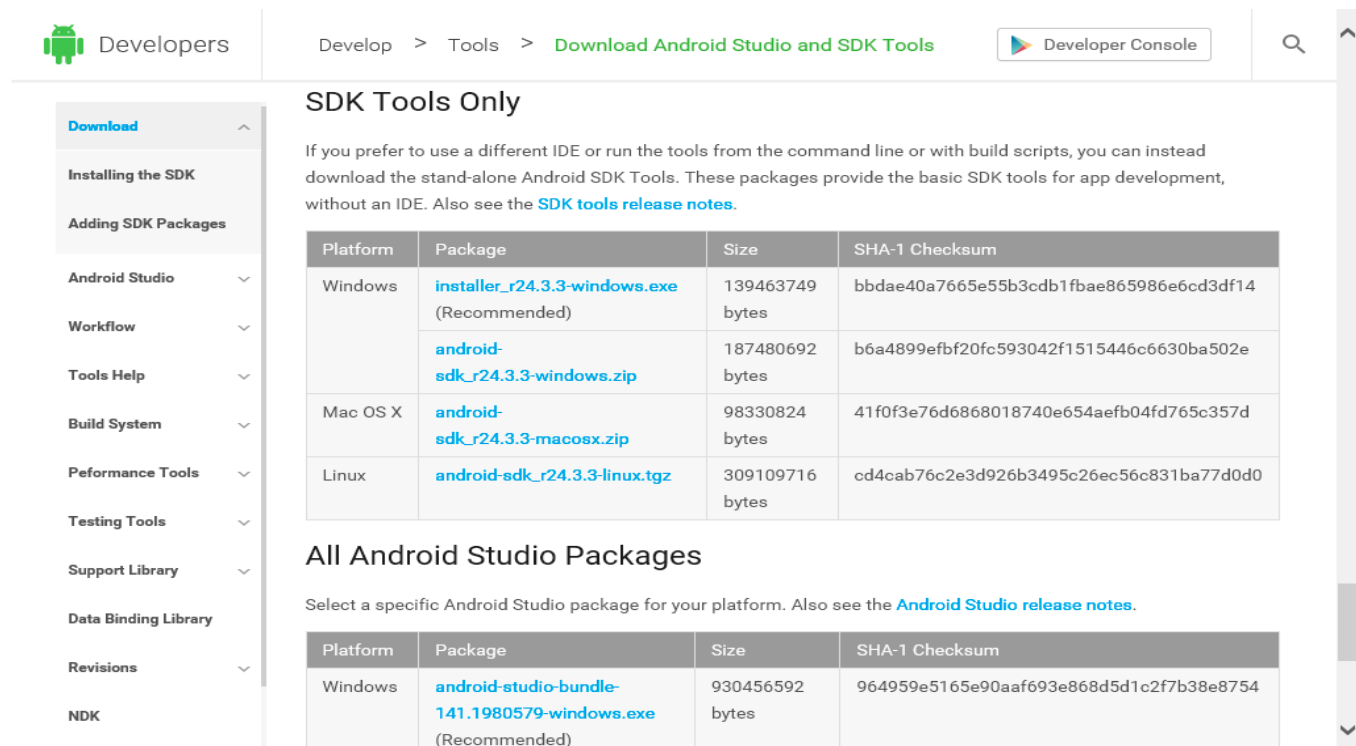
Select [STAND-ALONE SDK TOOLS] and click [Download the SDK now] in the page that is displayed next. The newest version of the SDK is r24.3.3as of August 2015. Download the installer (exe file) from here.

Run the downloaded installer to install the SDK.
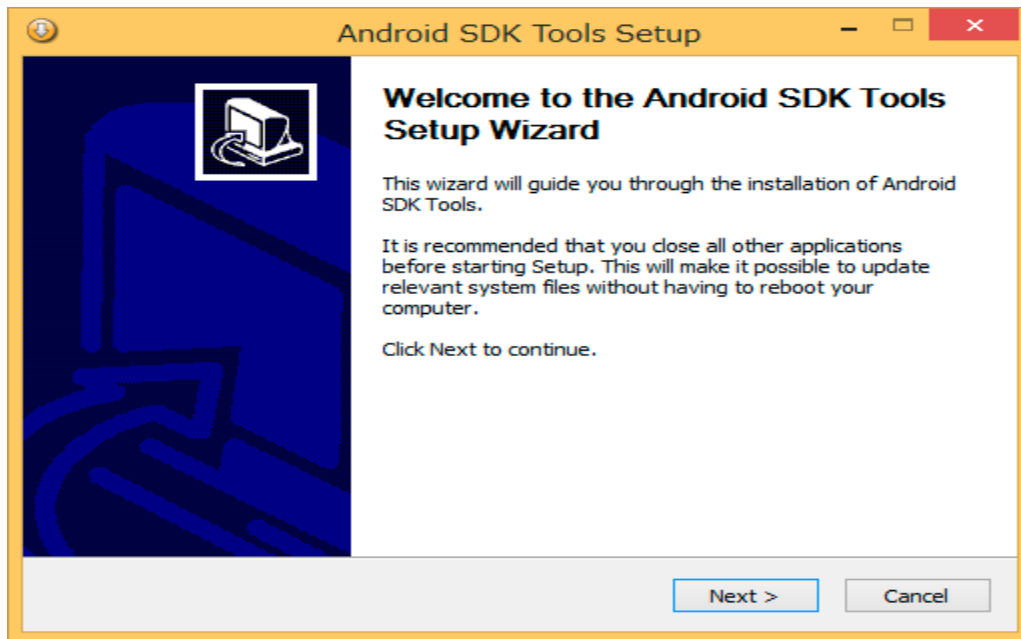
A screen where you will enter the location to install the SDK will eventually be displayed. Install directly to the C drive so that it is easy to find.
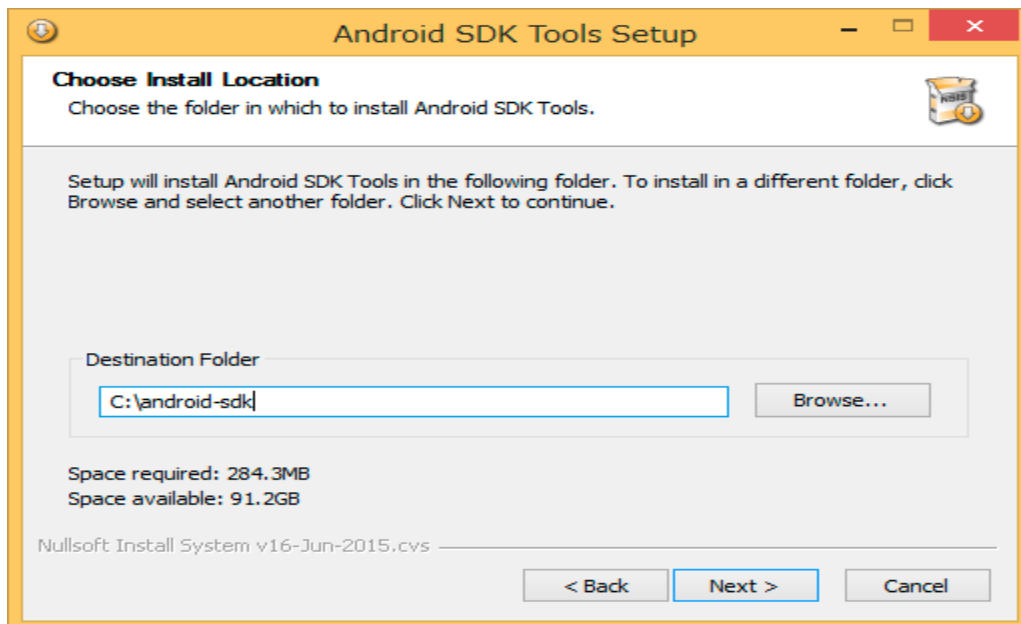
2-4-2 Downloading Additional Components Using Android SDK Manager
Only the bare essentials come with Android SDK right after installation. Here we will download some additional components such as platform-tools using Android SDK Manger. Android SDK Manger can be found in the start menu.

You can download necessary drivers for connecting devices to the host PC with USB through Android SDK Manager. If you're not sure which thing to download, you will get by with what is selected right after you run Android SDK Manager. However, be sure to select [Android SDK platform-tools], [Android SDK Build-tools] and the API (the newest version here is [Android 5.1.1 (API 22)]).
Also, it is probably a good idea to add [Google USB Driver] (mentioned later) in the [Extras] folder which makes it possible to install on to the device using USB.

- Android SDK platform-tools

- Android SDK Build-tools

- Android5.1.1 (API 22)

- Google USB Driver

*It is possible that the above tools are already selected on start up.
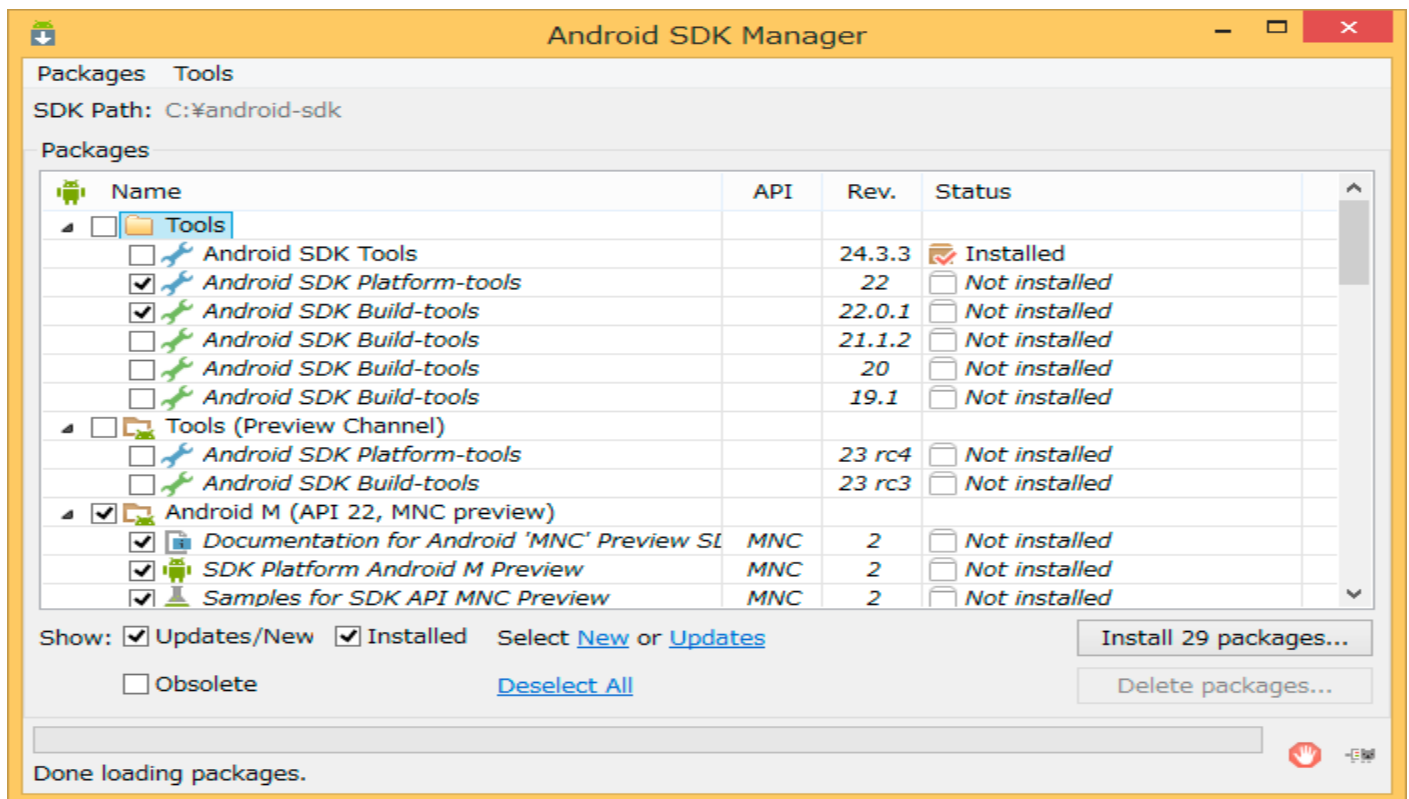
2-4-3 Path Setup & Check
Edit the System Environment Variable Path and add the location of Android SDK and the 3 locations for each tool.

```
;C:\android-sdk\

;C:\android-sdk\tools\

;C:\android-sdk\platform-tools\
```
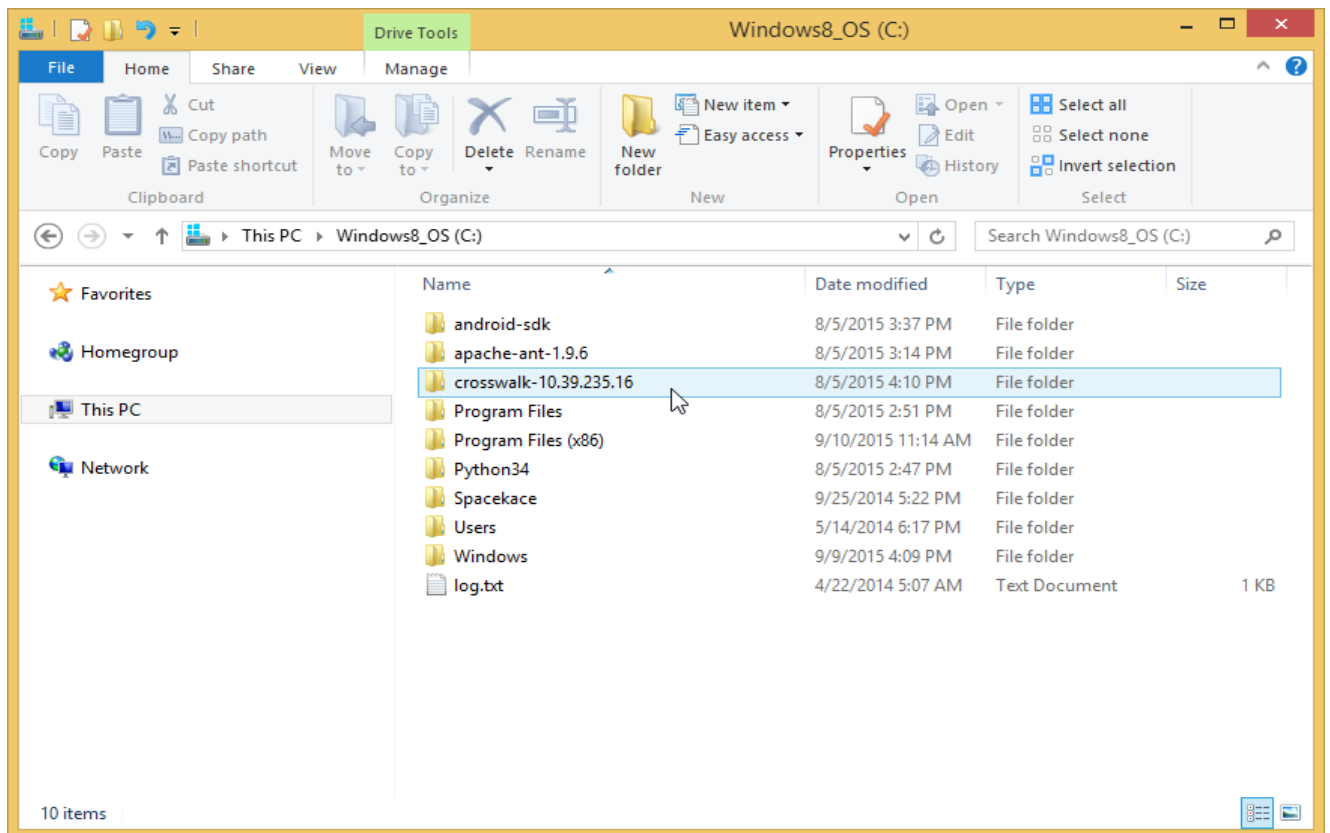
Check the Path. Let's try using the adb command here. Enter "adb version" into the command prompt and press the Enter key.

```
> adb version
```


2-5 Downloading the Android App Template for Crosswalk Project

You do not need to install Crosswalk Project. Extract the downloaded ZIP file wherever you like. Things such as the browser engine and packaging script are included inside the downloaded file.

Download the stable version of the Android app template from https://download.01.org/crosswalk/releases/crosswalk/android/stable/. Versions go all the way up to 14 but

10.39.235.16 is stable.
Click the folder and download [crosswalk-10.39.235.16.zip].

In this example, the file was extracted directly to the C drive.

With this, the packaging environment is now complete.

3 Deploying Games Using RPG Maker

Here we will export the necessary game data, assets and game engine to a package from RPG Maker. Select [Deployment...] from the File Menu and the deployment dialog will appear. If you select [Android / iOS] and export, all the files needed for Android packaging will be placed in the selected folder.
In this example, we deployed to the desktop.
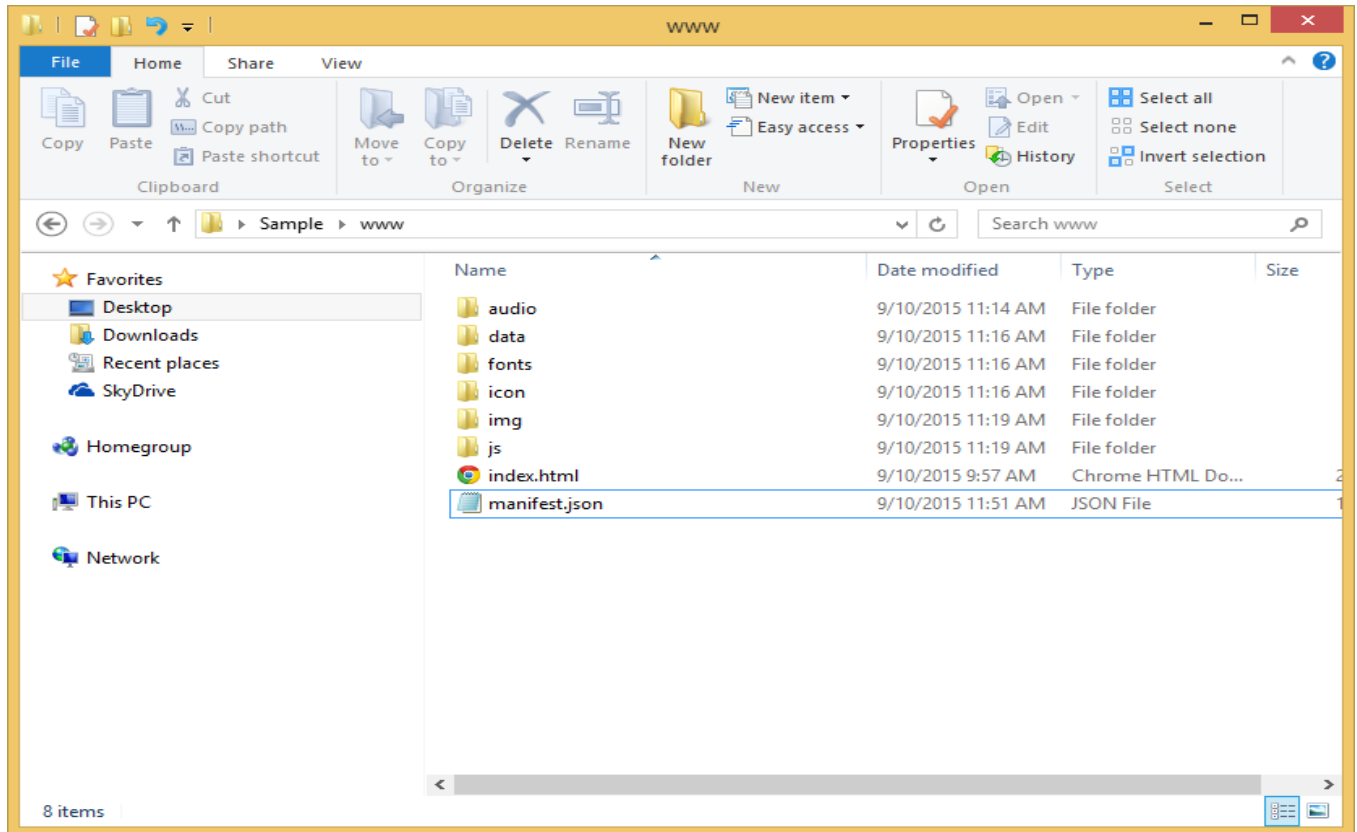
4 Packaging

Packaging refers to creating the apk file which you allow you to install applications to an Android device.

4-1 Creating manifest.json

Let's create the manifest.json file using your preferred text editor which determines the actions in your game.
Below is an example of a manifest.json file.

manifest.json

```
{
  "name": "Sample",

  "xwalk_version": "0.0.1",

  "start_url": "index.html",

  "display": "fullscreen",

  "orientation": "landscape",

  "icons": [

    {

      "src": "icon/icon48.png",
```

```
      "sizes": "48x48",
      "type": "image/png",
      "density": "4.0"
    },
    {
      "src": "icon/icon72.png",
      "sizes": "72x72",
      "type": "image/png",
      "density": "4.0"
    },
    {
      "src": "icon/icon96.png",
      "sizes": "96x96",
      "type": "image/png",
      "density": "4.0"
    },
    {
      "src": "icon/icon144.png",
      "sizes": "144x144",
      "type": "image/png",
      "density": "4.0"
    },
    {
      "src": "icon/icon192.png",
      "sizes": "192x192",
      "type": "image/png",
      "density": "4.0"
    },
    {
      "src": "icon/icon512.png",
      "sizes": "512x512",
      "type": "image/png",
      "density": "4.0"
    }
  ]
}
```

Aside from the name and version number, everything else can be left as is.
Save the manifest.json file in the "/www" folder which you deployed earlier.
The purpose of property names and values inside the file are as follows.

Property name	Value
name	The game's name.
xwalk_version	Version number. Increase the number if there is an update. Be sure to follow the X.X.X version format.
start_url	The game's URL. This will be set to index.html.
display	Specifies the full screen setting. This will be set to fullscreen.
orientation	The orientation of the screen. This will be set to landscape.
icons	Icons

Reference URL: Manifest for a web application: http://w3c.github.io/manifest/

4-2 Running the Packaging Script

Inside the Android app template that you downloaded in 2-5 is a file called "make_apk.py" which is a script that runs the packaging process. This is used through the command prompt.
Open the command prompt and move to the inside of the "/Game Name/www" folder (the below game folder) found in the folder where you deployed the game. It is more convenient if you open the game folder using explorer and open the command prompt from the file menu.
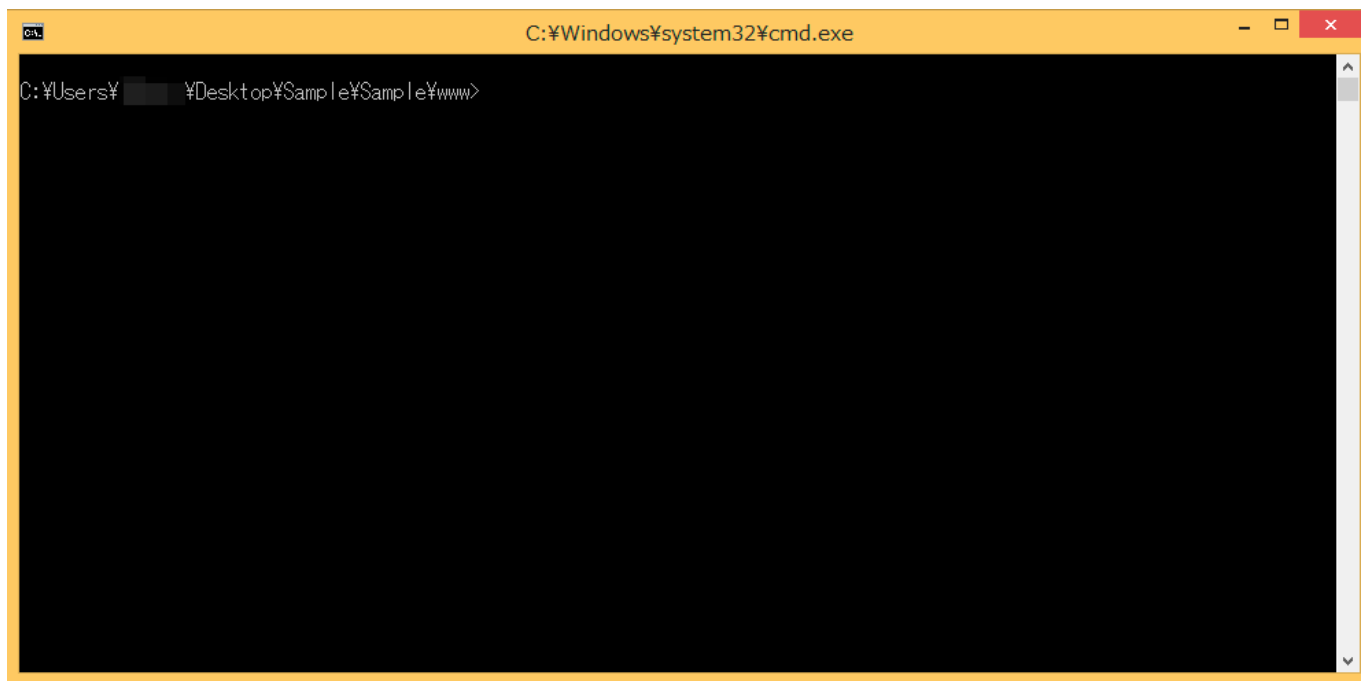
Below is an example of running the command on the environment we have just prepared.

```
python C:\crosswalk-10.39.235.16\make_apk.py --package=com.example.sample --manifest=manifest.json
```

The packaging script is written using Python. Using Python commands, we can specify the script.

--package

Specifies the package name. If you have a domain of your own, feel free to use it for your game. Just like the provided example, the domain name is used in reverse with the top level domain ("com" in the example) and is connected using a ".". Following at the end is your game's name ("sample" in the example"). You will use the package name used here forever for this game's package so please be careful.
* example.com is a domain that can be used temporarily. This cannot be published in the Play Store which will be explained later.
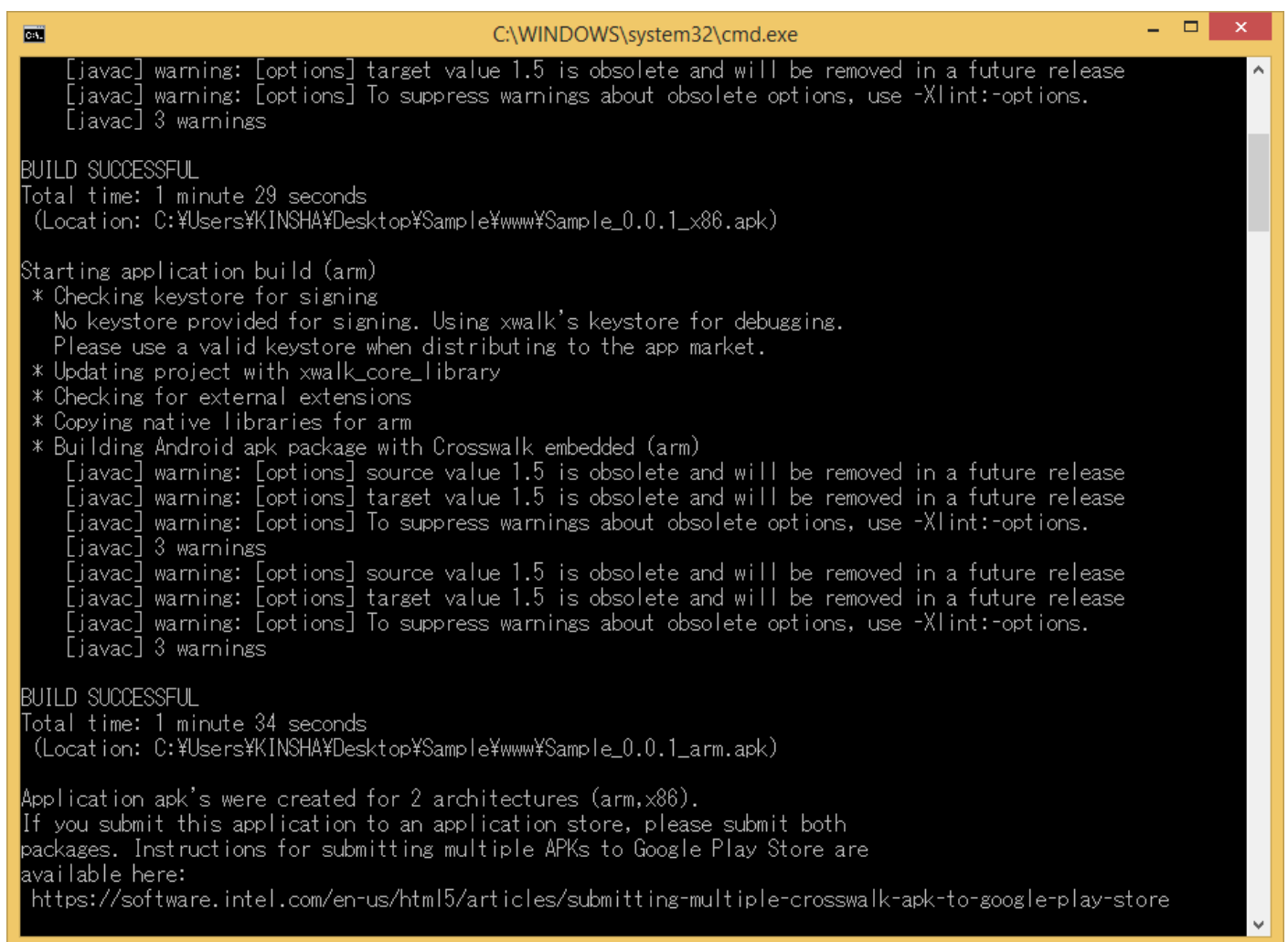
Domain example:
If the domain that you own is [example.com] and your game's name is [sample], you would write:
com.example.sample

--manifest

This specifies this package's manifest file. It is possible to specify things such as the game's version, icon and behavior related to the running of the game with the manifest file.

If you run the command and the packaging finishes without an issue, a [BUILD SUCCESSFUL] message will appear.

You can check that the two files, "Sample_0.0.1_arm.apk" and "Sample_0.0.1_x86.apk" have been created in the game folder. The file that contains [arm] is a package meant for the ARM architecture CPU which is a CPU widely used for implementation. This is used in the majority of Android smart phones and tablets currently in the market. The file containing [x86] is the package meant for the x86 architecture which is mainly used in computers.
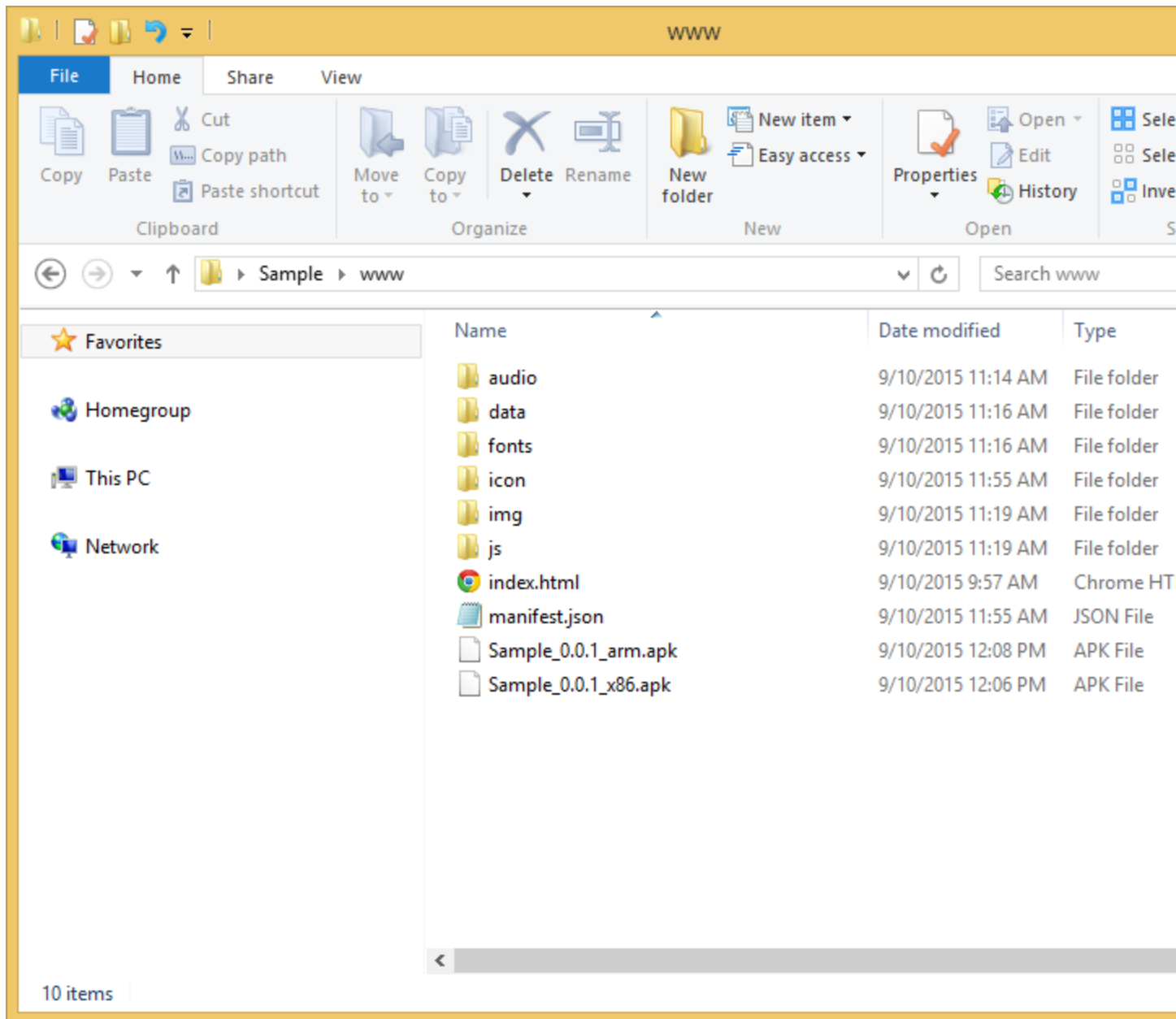
If you run the packaging script again, these two apk files will be included in the package so be sure to place the files outside of the folder.
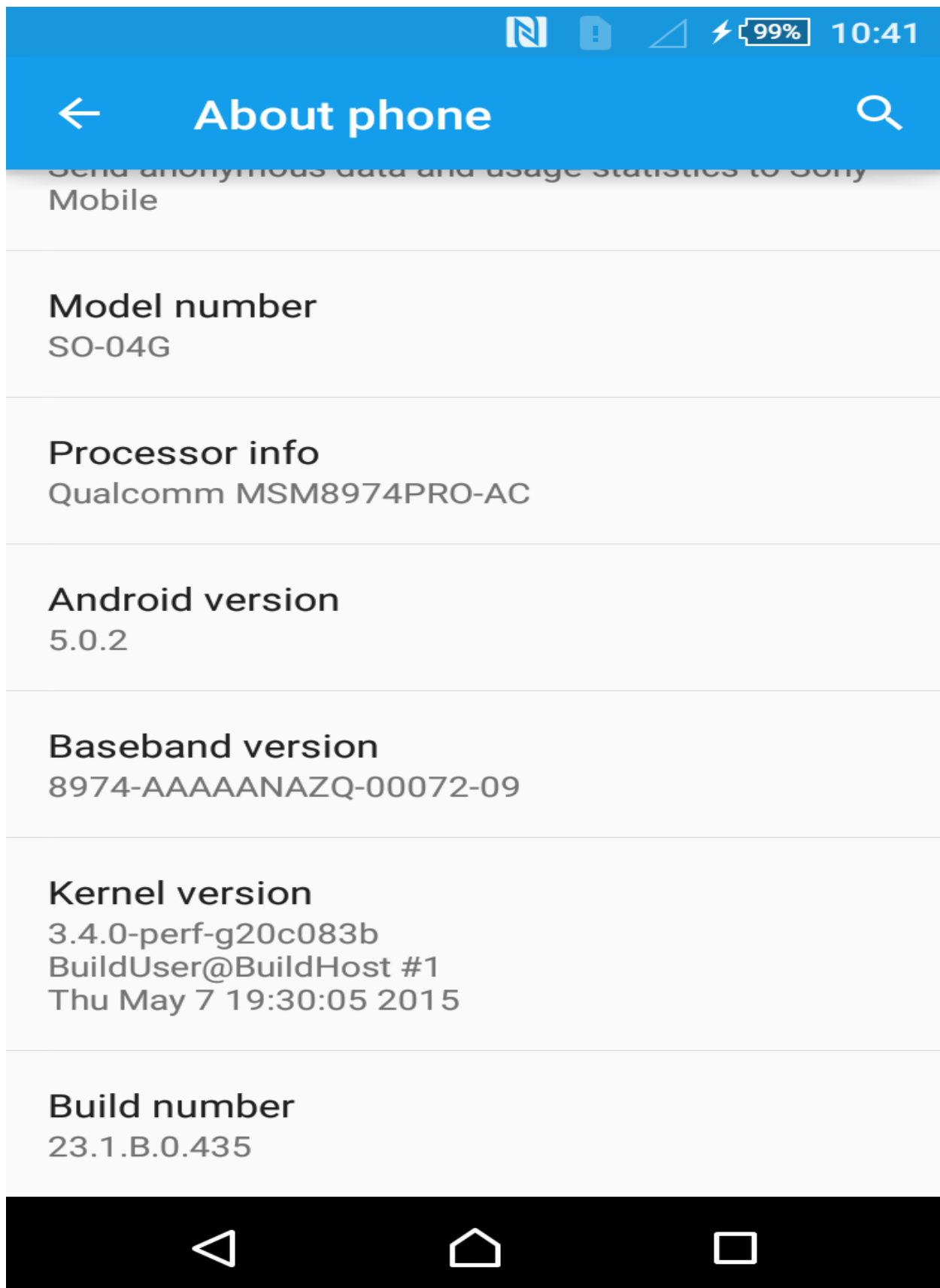
4-3 Installing to the Device

4-3-1 Preparing an Android Device

In order to be able to install a package using an USB connection, you need to enable [Developer Options] on your device.

In order to enable [Developer Options] go to [Settings] → [About] → [Software Information] → [More], and tap the [Build Number] text 7 times.

*The location of the [Build Number] may vary depending on the device you are using.

  Ex.) [Settings] → [About Device]

Send anonymous data and usage statistics to Sony Mobile

Model number
SO-04G

Processor info
Qualcomm MSM8974PRO-AC

Android version
5.0.2

Baseband version
8974-AAAAANAZQ-00072-09

Kernel version
3.4.0-perf-g20c083b
BuildUser@BuildHost #1
Thu May 7 19:30:05 2015
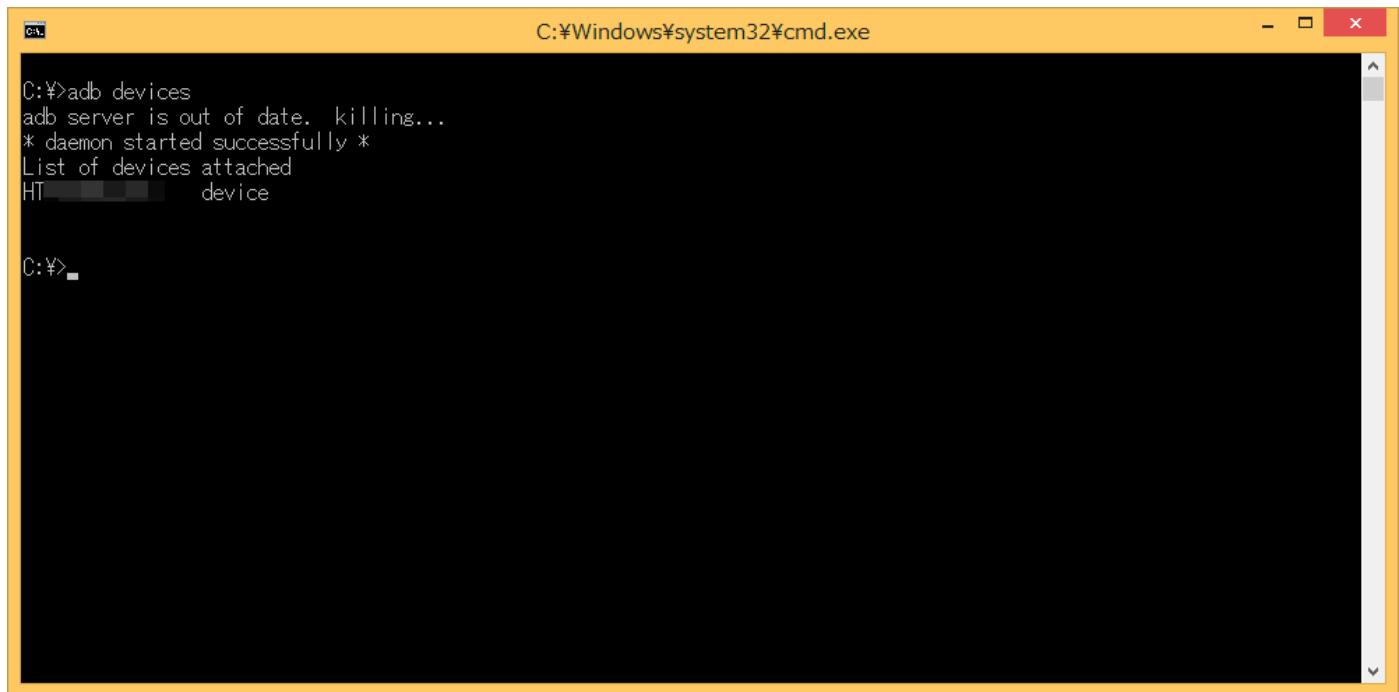
Build number
23.1.B.0.435

4-3-2 Installing Using an USB Connection
Here we will install the package using an USB connection. If you do not have the necessary USB drivers for your Android device on your computer, you can get them using the Android SDK Manager which was introduced in 2-4-3.

Connect the Android device to your PC using the USB cable. Be sure to use an USB cable that can be used for transferring data, not for recharging your device.
When successfully connect your device to your PC, a dialog will appear on your Android device asking you if you want to enable USB debugging so be sure to enable this.

Enter "adb devices" into the command prompt and press the Enter key. If you have managed to connect the device, the device's name will appear.

```
> adb devices
```


We will use the "install" option from the adb commands to install the package. Enter "adb install -r Sample_0.0.1_arm.apk" and press the Enter key. The "-r" option is an option used for reinstalling but if the name of the package is the same, you can update the game.
[Success] will be displayed once you've successfully installed the package. You will be able to run the game located in the list of applications on your device.

```
> adb install -r Sample_0.0.1_arm.apk
```
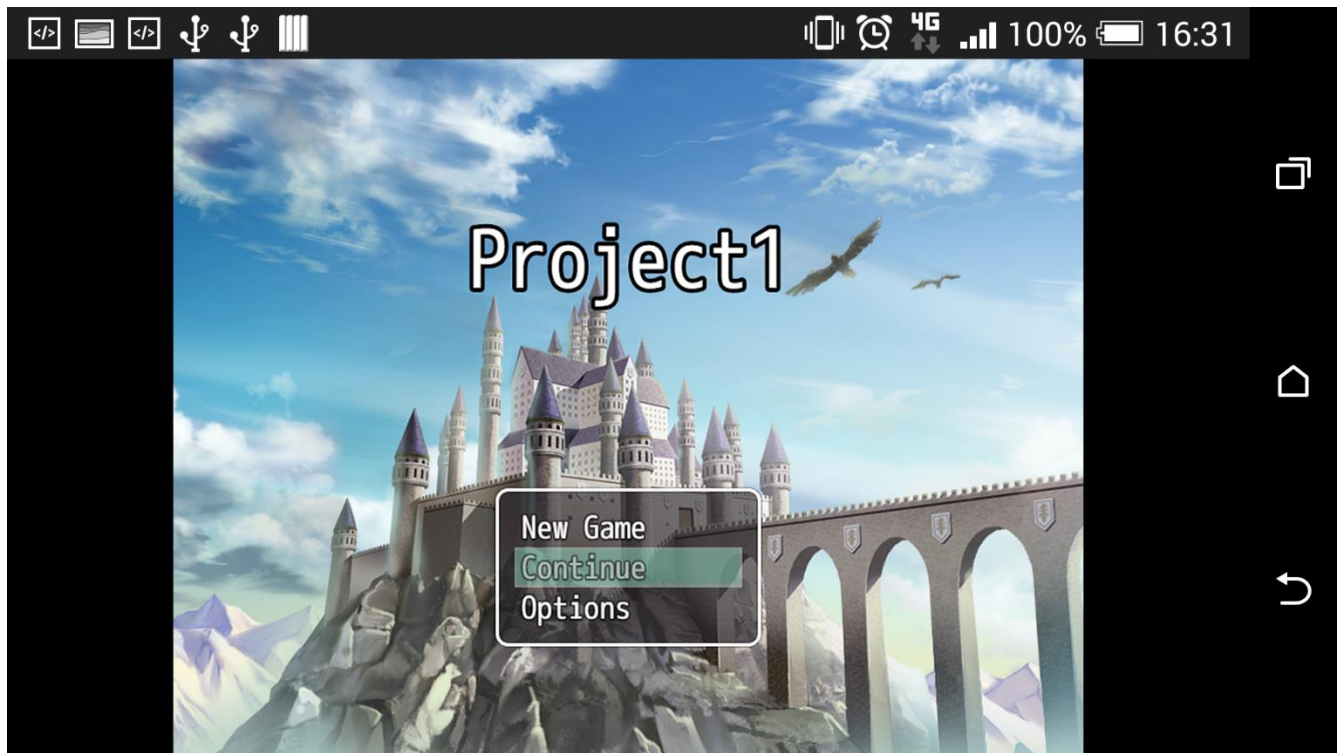
4-3-3 Installing via Internet

To use a cloud service such as Google Drive and installing via an Internet connection, you will need to upload the apk file to a cloud service of your choice and download and install using an Android device.

In order to do so you will need to set permissions to allow installations from unknown sources on the Android device.

Do this by navigating to [Settings] → [Security] → [Unknown sources] and set it this on.
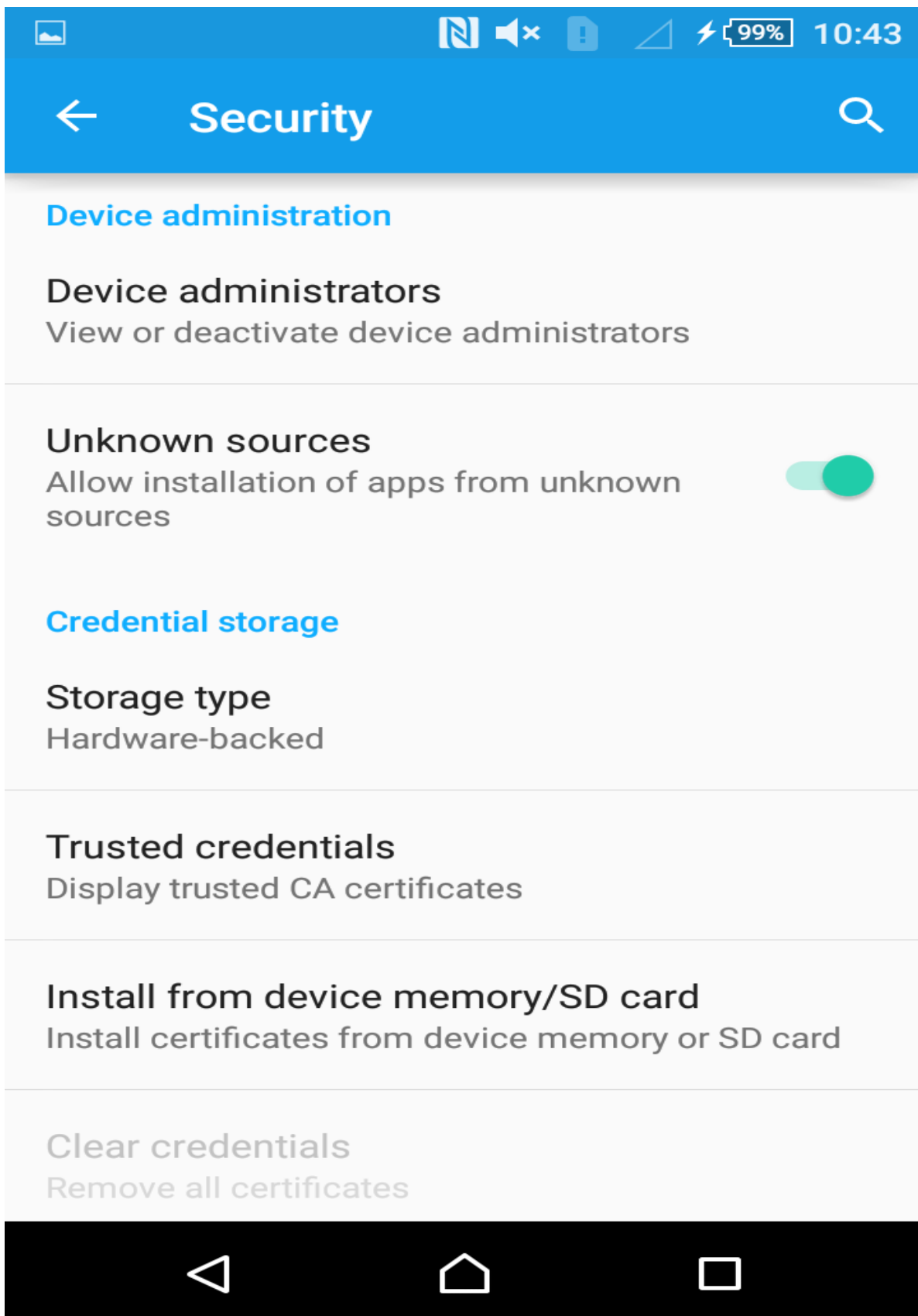
5 Publishing to Google Play

If you want to publish your game on Google Play, you will need to register on Google Play Developer.

We unfortunately cannot explain here all there is to know about app development for Google Play. It is our recommendation that you refer to reference books and other sources of information in the market.
Here we have written especially important information.

5-1 Signing an App

"Signing" an app refers to including proof in a package that the [Source], or in other words, you are the one that developed the game. This is similar to adding your signature to a piece of art which you have painted. Unsigned apps cannot be published on Google Play.
There are several ways to sign an app, but on this page we will introduce a way to sign your app using the make_apk.py script.

5-1-1 Creating a Keystore File

A keystore file is a type of file which contains your passworded information and prevents a third party from freely using your app. Through doing this, you can guarantee that you were the one that developed your game.
You can manage multiple keys on the [Store] but one is enough for an Android package.
Furthermore, you can use the keystore file created here on the signatures of your other apps as well.
We will use the keytool command included in the JDK which installed when setting up your environment to create the keystore file. Below is an example of a command.

```
keytool -genkey -v -keystore sample.keystore -storepass password -alias release -
keypass password -keyalg RSA -validity 10000
```

-genkey -v

Specifies to create a new key file.

-keystore
Specifies the name of the keystore file.

-storepass
Specifies the password for the store. In the example, we've used [password] but when actually creating a password, please set it to something that would be hard for a third party to figure out.

-alias

Specifies the alias of the key.

-keypass

Specifies the password of the key. In the example, we've used [password] but when actually creating a password, please set it to something that would be hard for a third party to figure out.
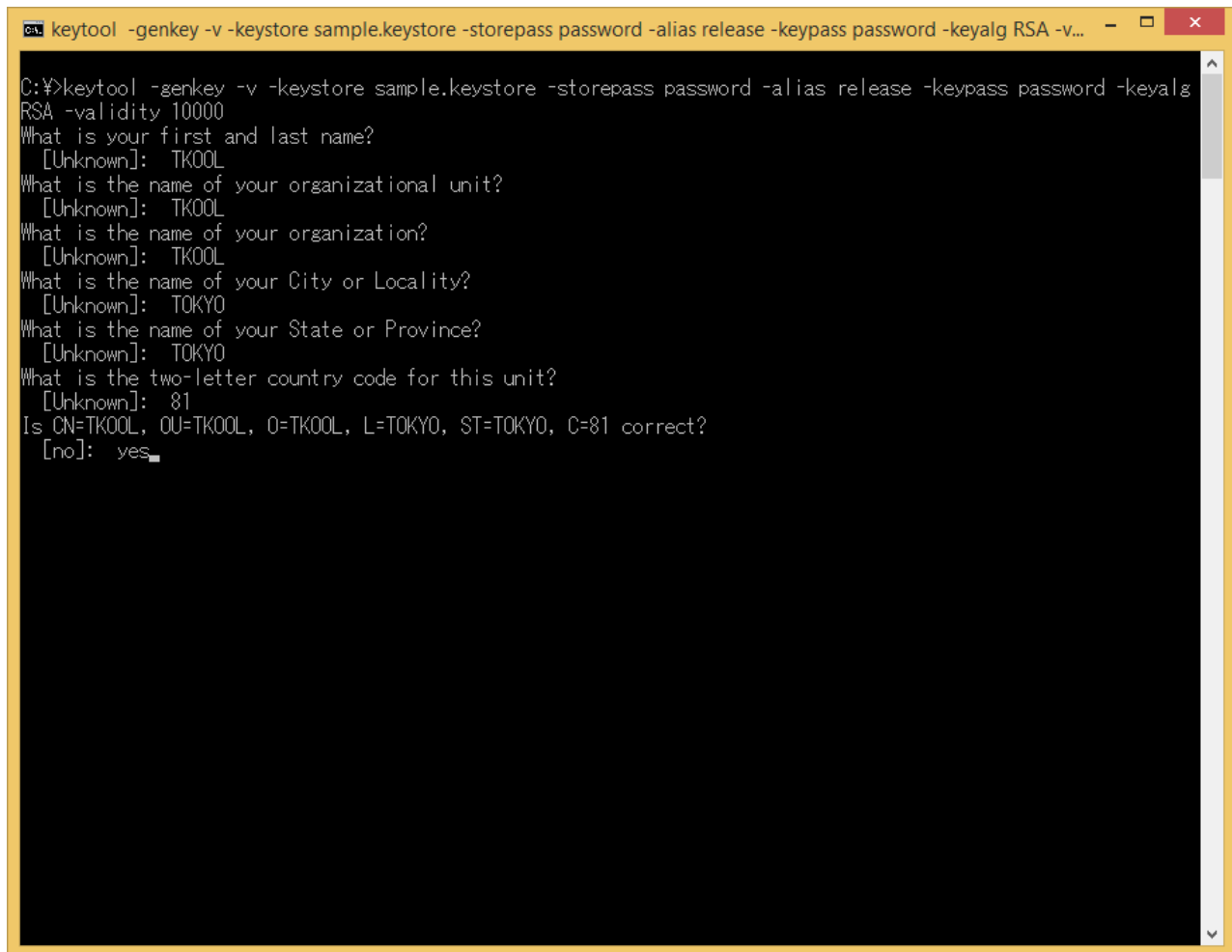
-keyalg

Specifies a passworded algorithm (here it is RSA).

-validity

Specifies the duration (in days) that the key file can be used. Set this to a long enough duration since the key file will continue to be used for things like updates.

After entering in a command and pressing the Enter key, you will be prompted to input the key information.

Please enter your own information that you have decided such as the name and organization name. Country code 81 refers to Japan.
If you are working in a <Japanese> environment, please enter <Yes> when asked <Are you sure?>.
A keystore file with the name "sample.keystore" will be created in the folder you opened using the command prompt.

5-1-2 Packaging Using a Keystore File
We'll use an additional option when running the packaging script mentioned in 4-1 to specify a keystore file for signing the package. Below is an example of using that additional option.

```
python C:\crosswalk-10.39.235.16\make_apk.py --package=com.example.sample --
manifest=manifest.json --keystore-path=C:\Users\IEUser\Desktop\sample.keystore --
keystore-alias=release --keystore-passcode=password --keystore-alias-passcode=password
```

The keystore-path, keystore-alias, keystore-passcode and keystore-alias-passcode options have increased.

--keystore-path

Using keystore-path specifies the location of the keystore file with a file name.

--keystore-alias

Specifies the alias of the keystore file when it was created.

--keystore-passcode

Specifies the password of the keystore file when it was created.

--keystore-alias-passcode

Specifies the key password of the keystore file when it was created.

Upload the apk file that you created to the Google Play Developer Console to apply. Please be sure to follow the instructions on the Google Play Developer Console for other necessary images, etc.
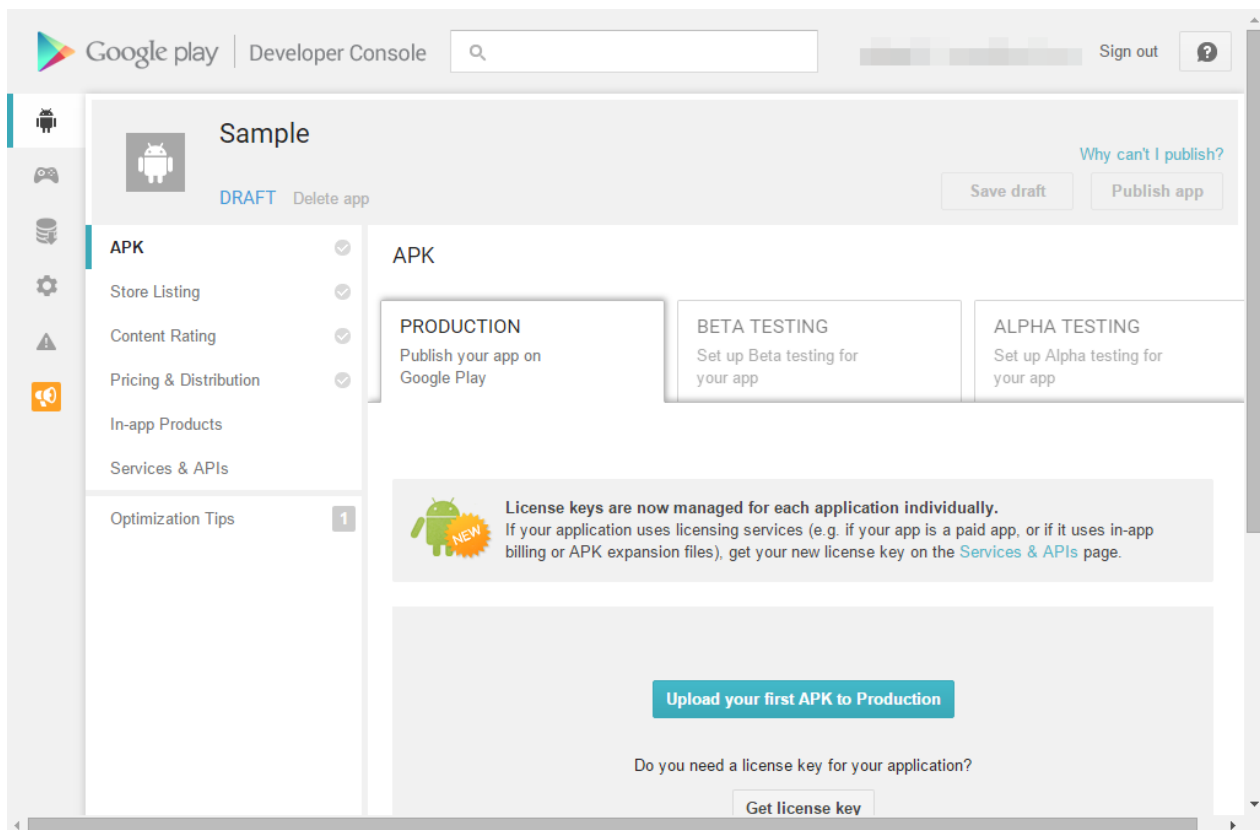
A Word of Caution

- We are not responsible for any issues that may occur on your PC using the environment settings and command controls.

- We cannot guarantee that all Android game apps that you can develop using the methods we have introduced here will be published on Google Play.

- This information is current as of August 2015. Some of the images displayed here may differ as all the software used here is constantly being updated.

- If you have any questions regarding [App Submission to Google Play], please refer to [Google Play Developer Help](#) .