Readme

Windows Azure Toolkit for Android

* 1. Document version: 1.0.0
  2. Last updated: 12/2/2011
  3. ****

# Contents

[Setting up a developer environment 3](#_Toc310599165)

[Software Requirements 3](#_Toc310599166)

[Installing Eclipse 3](#_Toc310599167)

[Downloading the ADT Plugin 4](#_Toc310599168)

[Installing and configuring the Android SDK 5](#_Toc310599169)

[How to configure proxy settings in Eclipse 13](#_Toc310599170)

[Getting started with the toolkit & using bits on GitHub 15](#_Toc310599171)

[Importing the Windows Azure Toolkit for Android project from GitHub 15](#_Toc310599172)

[Building the projects 21](#_Toc310599173)

[Running the Tests 23](#_Toc310599174)

[Troubleshooting 24](#_Toc310599175)

[Using the sample application 29](#_Toc310599176)

[Introduction 29](#_Toc310599177)

[Configuring the application using Windows Azure credentials (Direct-Mode) 29](#_Toc310599178)

[Configuring the application using the Proxy Services with Membership 30](#_Toc310599179)

[Configuring the application using the Proxy Services with Access Control 31](#_Toc310599180)

[Running the sample application 32](#_Toc310599181)

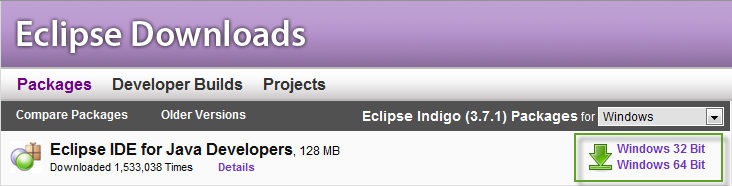
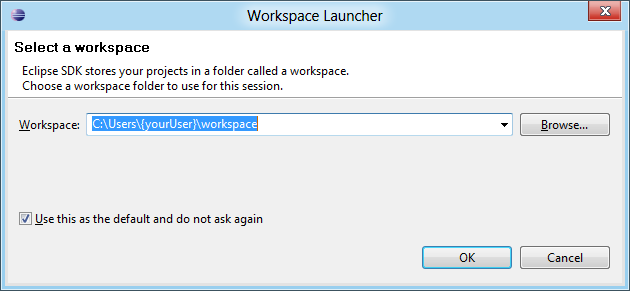
Setting up a developer environment

* 1. This section describes the software requirements for running this toolkit and systematic instructions for the installation.

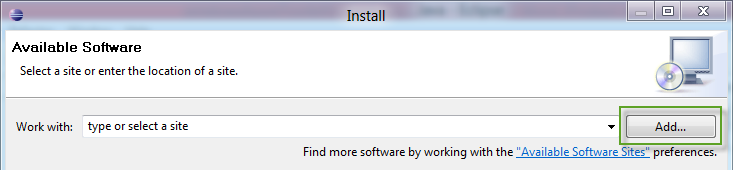
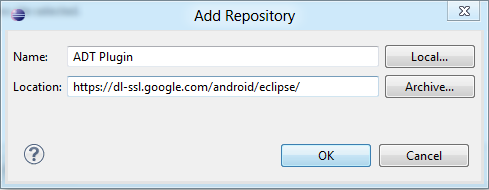
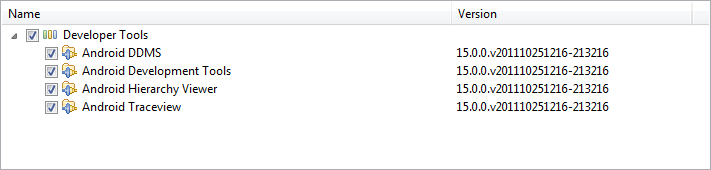
Software Requirements

* + Eclipse Indigo IDE for Java Developers.
  + Android Development Tools (ADT) plugin for the Eclipse IDE.
  + Android SDK 2.2 or above except the 2.3.3 which introduces a bug on the JavaScript to java bridge, more information can be found [here](http://code.google.com/p/android/issues/detail?id=12987).

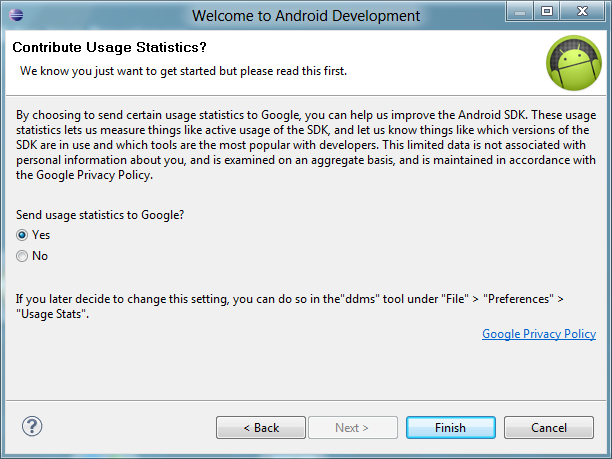
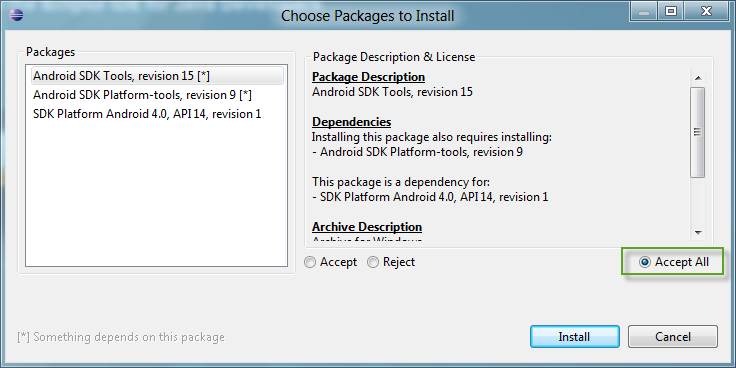
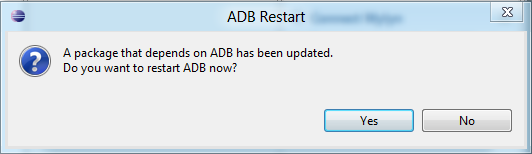
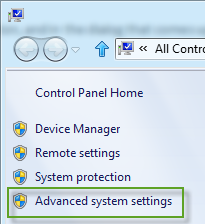
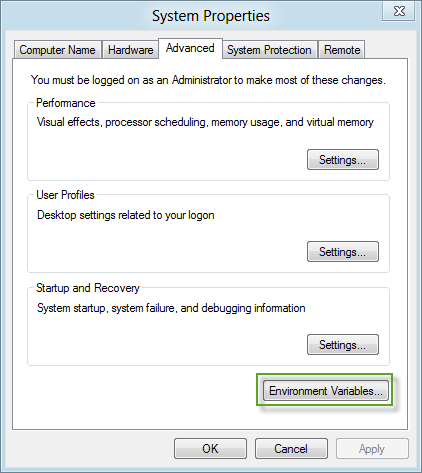
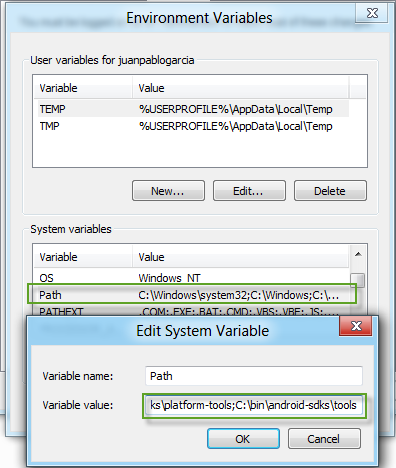
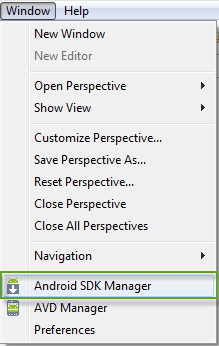
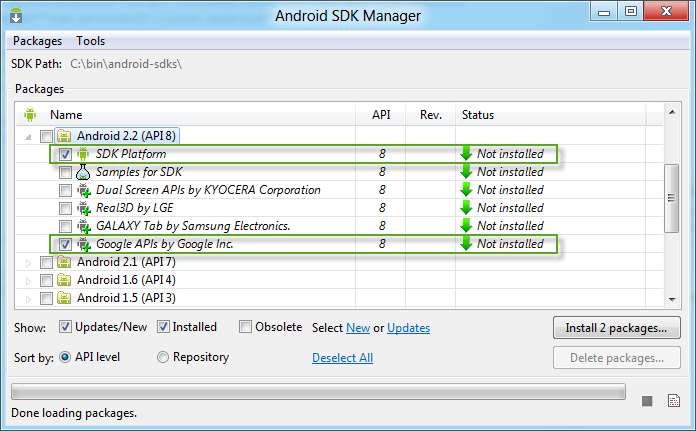
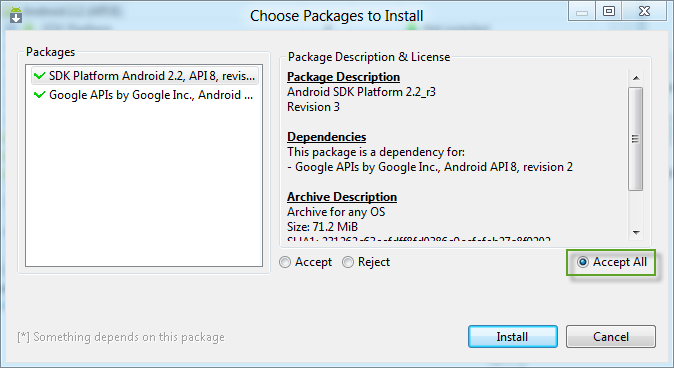
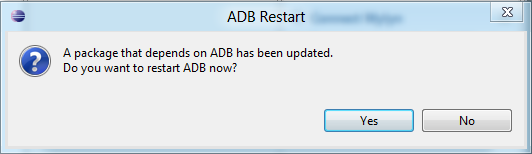
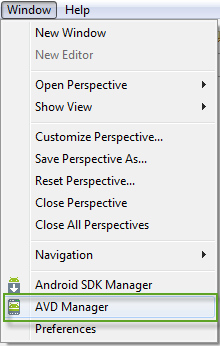
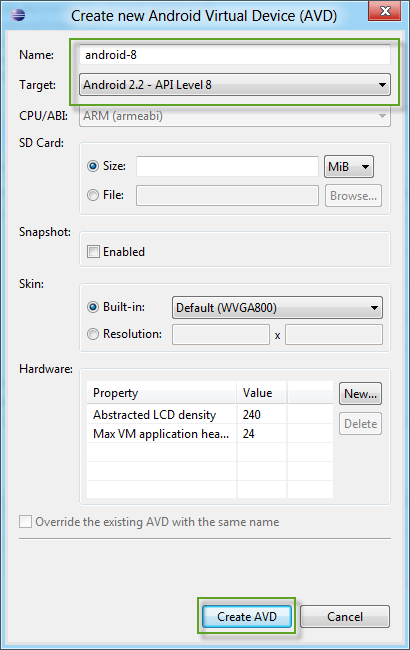
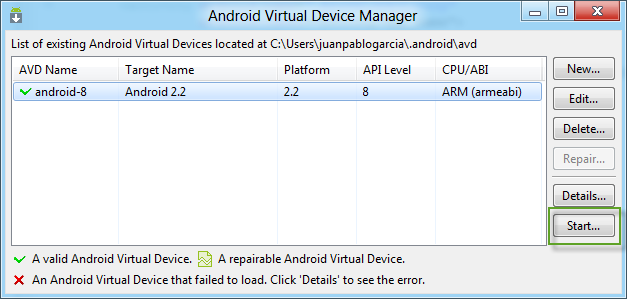
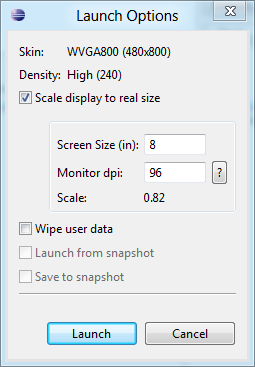
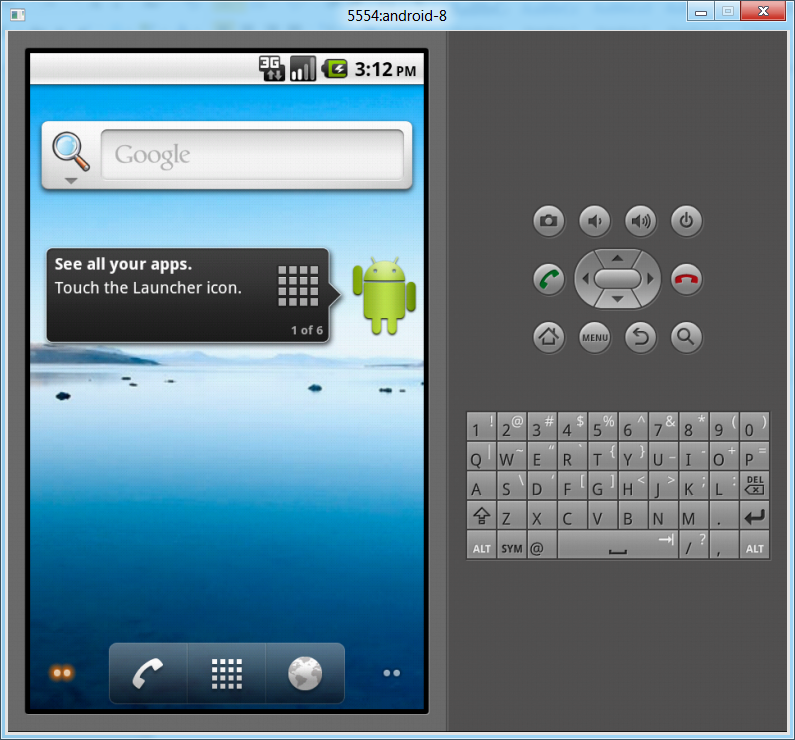
Installing Eclipse

* 1. Download and Install the Java JDK from <http://www.oracle.com/technetwork/java/javase/downloads/jdk-7u1-download-513651.html>
  2. Download Eclipse IDE for Java Developers from <http://www.eclipse.org/downloads/>
     1. 
  3. Extract the eclipse folder wherever you want (e.g. c:\eclipse).
  4. Go to the folder and run **eclipse.exe.**
  5. Choose the workspace location e.g. C:\Users\{yourUser}\workspace and click on the *“Use this as the default and do not ask again”* option.
     1. 

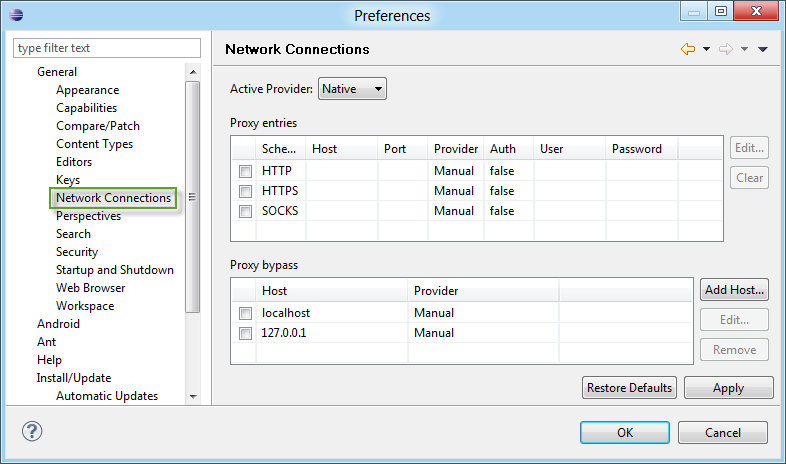
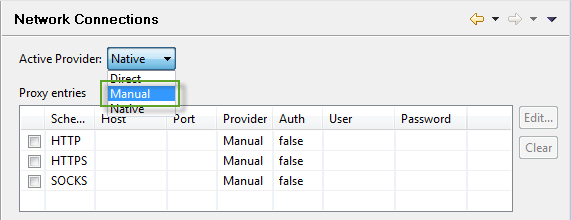
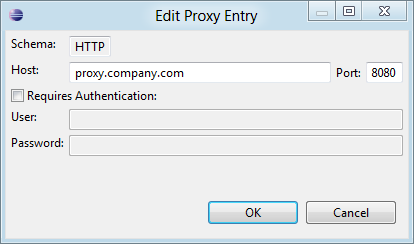
Downloading the ADT Plugin

* 1. Use the Update Manager feature of your Eclipse installation to install the latest revision of ADT on your development computer.
  2. Assuming that you have a compatible version of the Eclipse IDE installed, as described in [Installing Eclipse](#_Installing_Eclipse), above, follow these steps to download the ADT plugin and install it in your Eclipse environment.
  3. Start Eclipse, then select **Help** -> **Install New Software...** from the menu.
  4. Click **Add**, in the top-right corner.
     1. 
  5. In the Add Repository dialog that appears, enter "ADT Plugin" for the Name and the following URL for the Location:
     1. https://dl-ssl.google.com/android/eclipse/
     2. 
  6. Click **OK.**
     1. **Note**: If you have trouble acquiring the plugin, try using "http" in the Location URL, instead of "https" (https is preferred for security reasons) or if you are behind a proxy server you can follow the steps described in the [How to configure proxy settings in eclipse](#_How_to_configure) section.
  7. In the Available Software dialog, select the checkbox next to Developer Tools and click **Next**.
     1. 
  8. In the next window, you will see a list of the tools. Click **Next**.
  9. Read and accept the license agreements, then click **Finish**.
     1. **Note**: If you get a security warning saying that the authenticity or validity of the software cannot be established, click **OK**.
  10. When the installation completes, restart Eclipse.

Installing and configuring the Android SDK

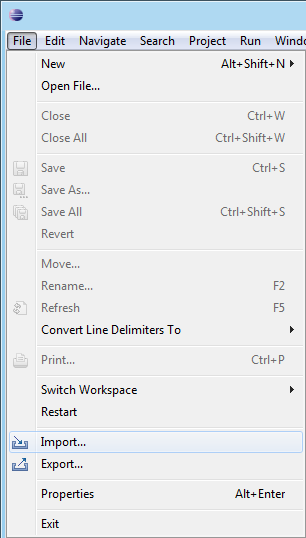
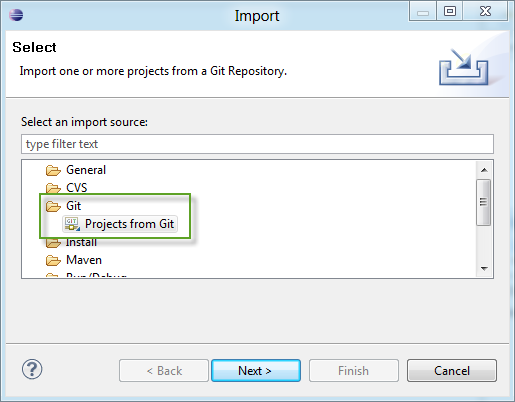
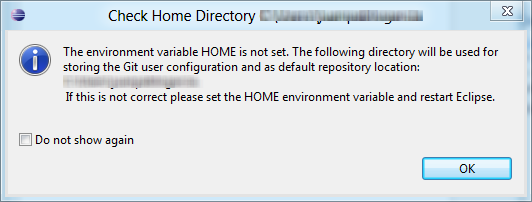
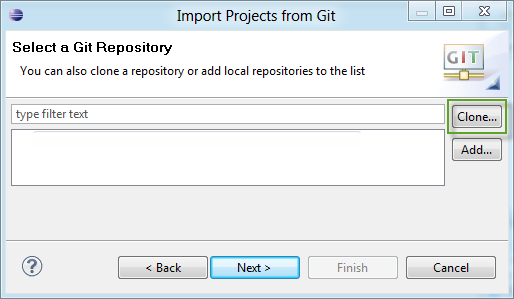
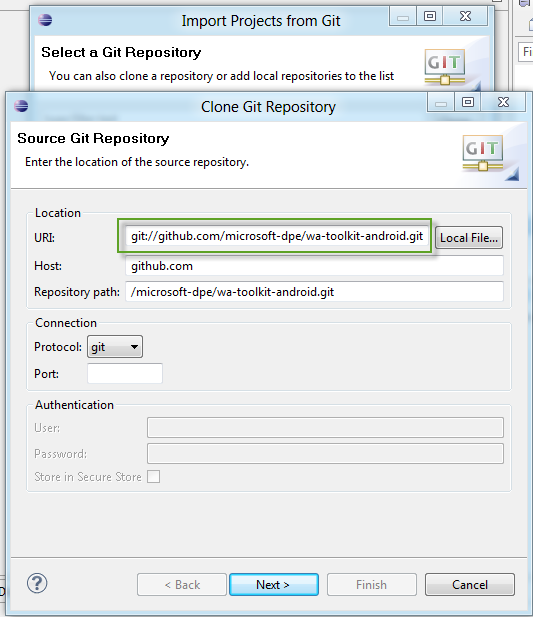
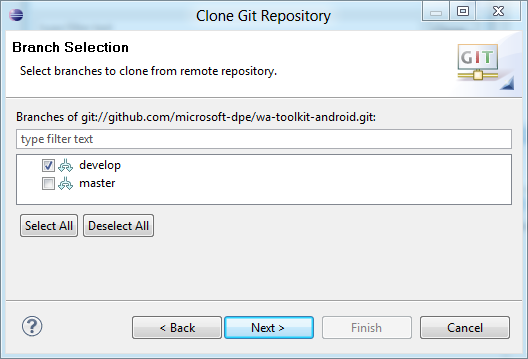
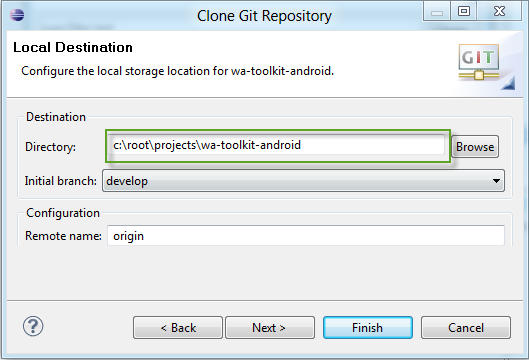
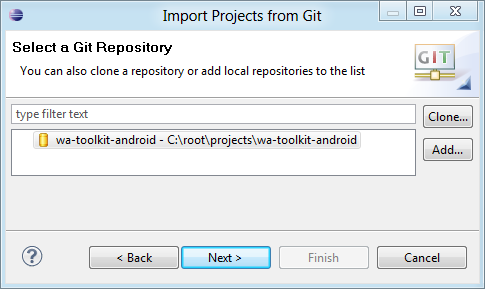
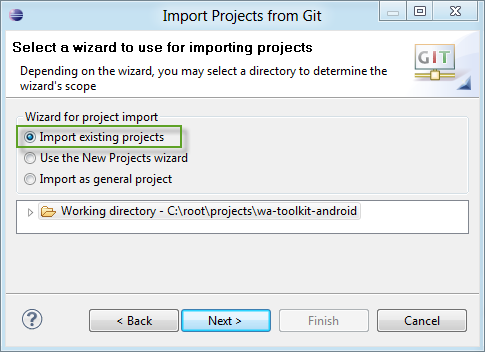
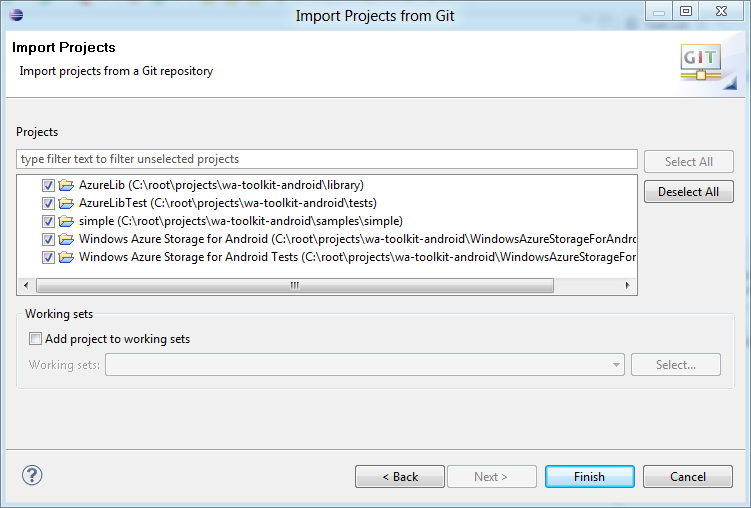
* 1. After restarting Eclipse, a new window will appear to configure the SDK, leave the default options and provide the folder where the SDK will be installed.
     1. 
     2. **Note**: It is recommended to choose a location without blank spaces, since there were known issues in previous versions.
  2. Choose if you are going to contribute with Google’s statistics.
     1. 
  3. Select the **Accept All** option for package descriptions and licenses and click the **Install** button.
     1. 
     2. **Note**: It takes a couple of minutes to start the installation.
  4. Restart **ADB**.
     1. 
  5. Go to **My Computer** andright-click **Properties**.
  6. Click the **Advanced system settings** link.
     1. 
  7. Under the **Advanced** tab, click the **Environment Variables** button, and in the dialog that comes up, double-click **Path** (under System Variables).
     1. 
  8. Add the full path to the **tools** and **platform-tools**, those folders are located inside the destination you provided when installing the SDK (e.g. c:\bin\android-sdks\tools & c:\bin\android-sdks\platform-tools).
     1. 
  9. Switch back to Eclipse.
  10. Open the **Android SDK Manager** from **Window** menu.
      1. 
  11. Since the project is targeted to the **API 8**, select from the **Android 2.2 (API 8)** node only the **SDK Platform** and **Google APIs by Google Inc.** items, then click the **Install 2 packages…** button.
      1. 
  12. **Accept All** package descriptions & licenses when prompted and click **Install.**
      1. 
  13. Restart **ADB**.
      1. 
  14. Close the Log window and the **Android SDK Manager**.
  15. Open the **AVD Manager** from the **Window** menu item.
      1. 
  16. Click the **New…** button to create a new virtual device targeted to the API 8.
  17. Provide a valid name; select the **Android 2.2 – API Level 8** from the drop-down list and click the **Create AVD** button.
      1. 
  18. Start the virtual device to check that it is working fine.
      1. 
  19. If you want, you can scale the device to the real size.
      1. 
  20. This is how the virtual device should look like.
      1. 
      2. **Note**: It may take a few minutes to load.
  21. You can find more information about Android development in Android’s Developers website at <http://developer.android.com/index.html>

How to configure proxy settings in Eclipse

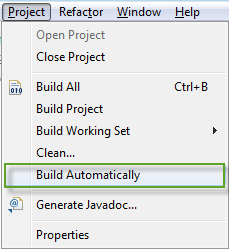
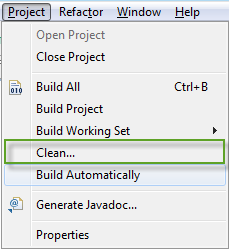
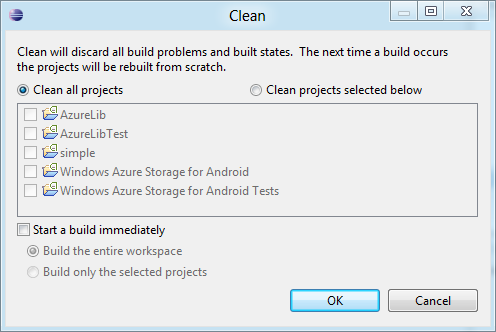
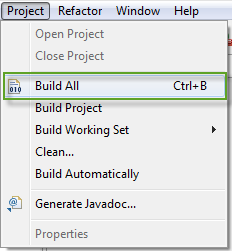
* 1. If your computer is behind a proxy server, you should follow these steps.
  2. Open the **Window** -> **Preferences** menu option.
  3. Select **Network Connections** on the left pane.
     1. 
  4. Choose **Manual** from the drop-down list.
     1. 
  5. Select **HTTP** in the list, click the Edit button, and provide your proxy configuration.
     1. 
  6. Click **OK**.
  7. If you want to access to an https address you should perform the steps described above but clicking on **HTTPS** instead.
  8. Click **OK** on the **Preferences** window.

Getting started with the toolkit & using bits on GitHub

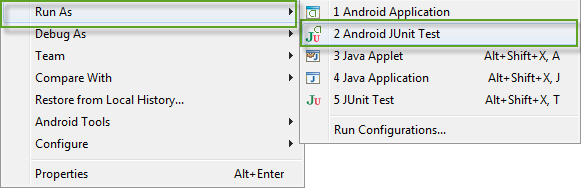
Importing the Windows Azure Toolkit for Android project from GitHub

* 1. Open Eclipse and choose **File** -> **Import** from the menu.
  2. Locate the **Git** node and choose *“Projects from Git”* and click **Next**.
     1. 
  3. Click **OK** on the **Home Directory** warning.
     1. 
  4. Click **Clone** on “Import project from Git” window.
     1. 
  5. Provide the repository location on the URI field, in this case:
     1. git://github.com/microsoft-dpe/wa-toolkit-android.git
     2. 
  6. Click **Next.**
  7. Choose the branch you want to clone.
     1. 
  8. Click **Next.**
  9. Choose your local destination and click **Finish.**
     1. 
  10. Once the repository has been cloned, click the **Next** button.
      1. 
  11. Select **Import existing projects** and click **Next.**
      1. 
  12. Choose the projects you want to install from the list and click **Finish.**
      1. 

Building the projects

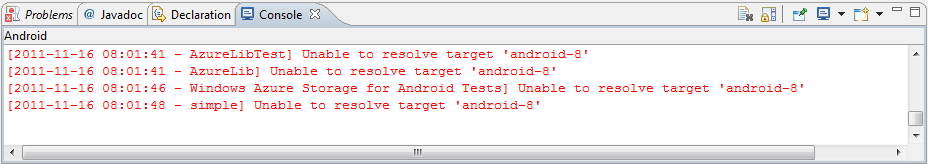
* 1. The following section describes how you can build and run the applications and tests included. If you encounter some issues, please refer to the [Troubleshooting](#Troubleshooting) section on this document.
  2. Remove the **Build Automatically** check from the **Project** menu.
     1. 
  3. From the **Project** Menu select **Clean….**
  4. Select **Clean all projects** and remove the “*Start a build immediately”* option.
     1. 
  5. Click **OK.**
  6. Finally, from the **Project** menu select **Build All.**

Running the Tests

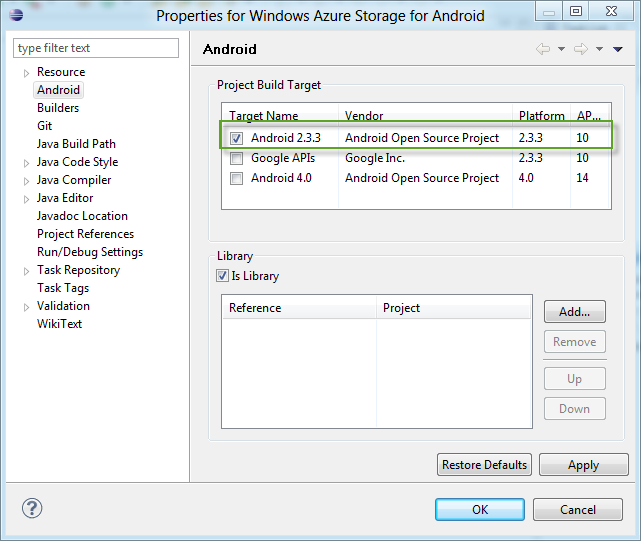
* 1. Right Click on the test project and select **Run As** -> **Android JUnit Test**.
     1. 

Troubleshooting

#### Symptom

* 1. You receive a warning regarding the target version.
  2. 

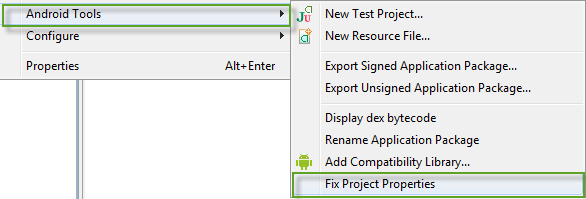
#### Solution

* 1. Right-click each project and select **Properties -> Android**. Then select the **Android 2.3.3** version as the project build target.
     1. 

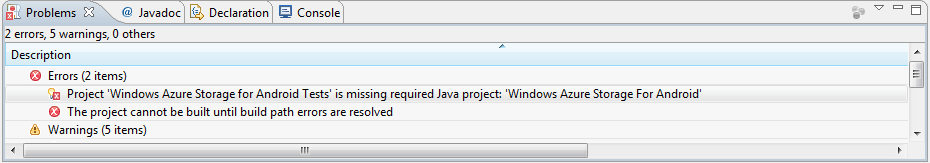
#### Symptom

* 1. You receive the ***‘Android requires compiler compliance level 5.0 or 6.0. Found '1.4' instead. Please use Android Tools > Fix Project Properties.’*** warning.

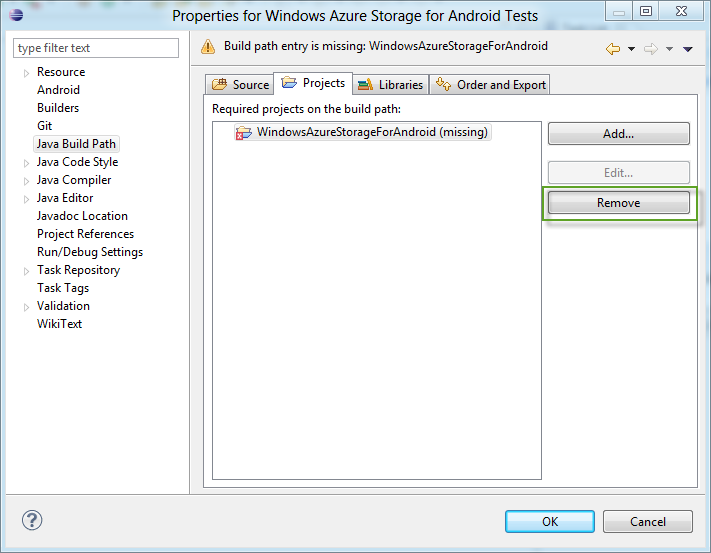
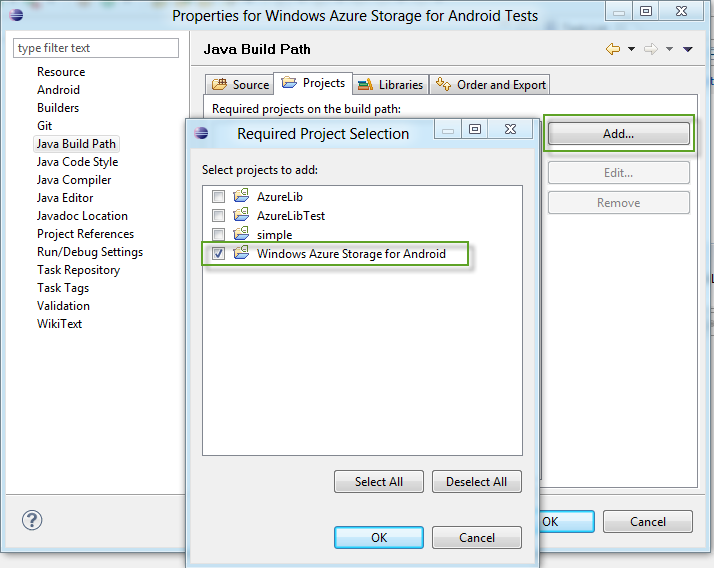
#### Solution

* 1. Locate the projects with warnings on the Package Explorer and right-click **Android Tools -> Fix Project Properties.**
     1. 

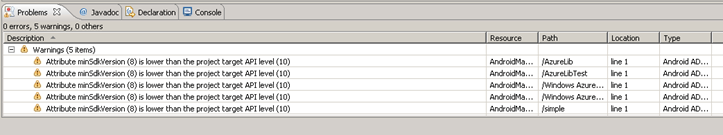
#### Symptom

* 1. You receive a **“project ‘A’ is missing required Java project: ‘B’”** error.
  2. 

#### Solution

* 1. Right-click the project with missing references and choose **Properties.**
  2. Click the **Java Build Path** item on the left menu, go to the **Projects** tab, select the missing project and click **Remove** button.
     1. 
  3. Click **Add…** and select the missing project from the list.
     1. 

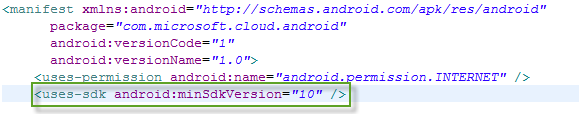
#### Symptom

* 1. You receive several warnings regarding **minSdkVersion**.
  2. 

#### Solution

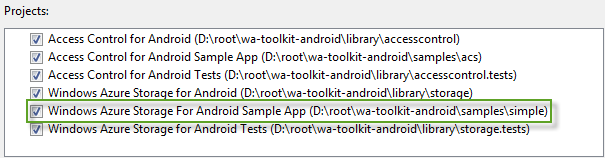
* 1. Double-click each warning; it will open the **AndroidManifest.xml** file.
  2. Click on the **AndroidManifest.xml** label to display the manifest in raw XML format.



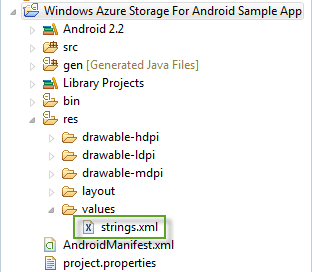
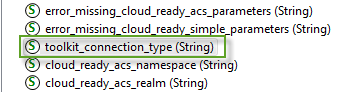
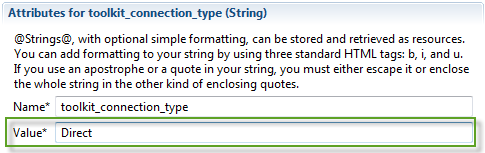
* 1. Update the **android:minSdkVersion** to **10.**
     1. 

Using the sample application

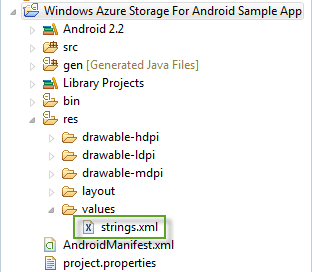
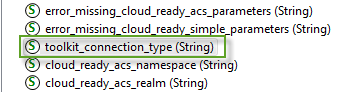
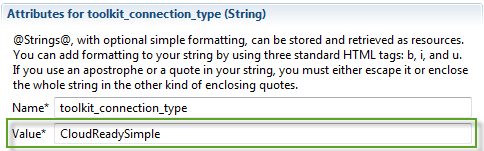
Introduction

* 1. The following section describes how to configure and run the sample application included as part of the Windows Azure Toolkit for Android.
  2. **Sample Location**
  3. The sample application’s source code is located at the **$/samples/simple** folder on the *master* branch on **GitHub**.
  4. If you followed the steps described in the “[Getting started with the toolkit & using bits on GitHub](#GettingStarted)” section and while on the import step you selected the simple folder, the sample should be already there in your workspace.
  5. 
  6. **Authentication Mechanisms**
  7. Before running the application you must select which authentication method will be used to consume the Windows Azure Storage Services. The toolkit works in two ways – the toolkit can be used to access Windows Azure storage directly, or alternatively, can go through a proxy server. The proxy server code is the same code as used in the [Windows Azure Toolkit for Windows Phone](http://watwp.codeplex.com/) and negates the need for the developer to store the Windows Azure storage credentials locally on the device. If you are planning to test using the proxy server, you will need to download and deploy the services found in the [wa-toolkit-cloudreadypackages](https://github.com/microsoft-dpe/cloudreadypackages) on GitHub.
  8. If you go for the proxy services, you can use either **Membership** or **Access Control Service** authentication mechanism.

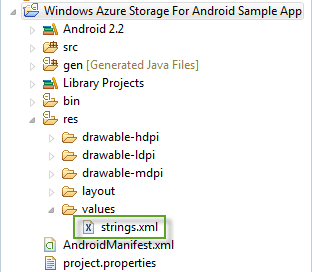
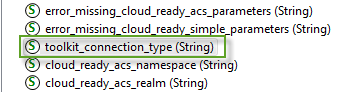
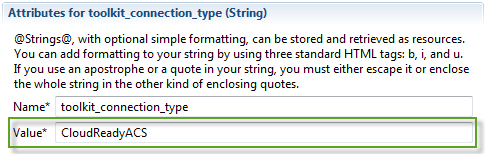
Configuring the application using Windows Azure credentials (Direct-Mode)

* 1. Navigate the **res/values** folder inside the **Windows Azure Storage for Android Sample App** project and double-click the **strings.xml** file.
     1. 
  2. Locate and click the **toolkit\_connection\_type** entry.
     1. 
  3. On the right pane, change the placeholder text by ‘**Direct**’ value.
     1. 
  4. Perform the same steps described above to provide the Windows Azure account information:
     1. **direct\_account\_name:** The storage account name.
     2. **direct\_access\_key**: The storage account access key.

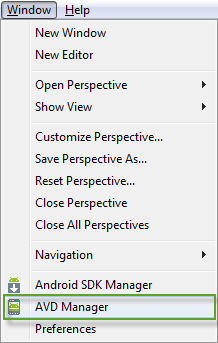
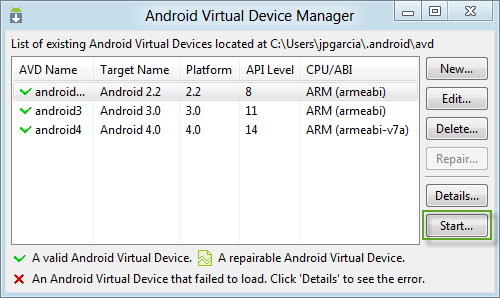
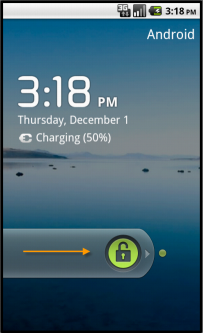
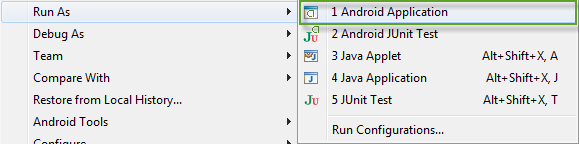
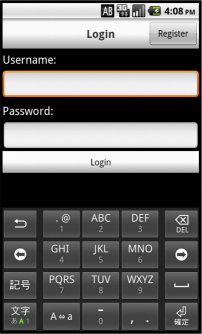
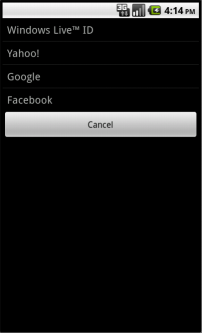
Configuring the application using the Proxy Services with Membership

* 1. Navigate the **res/values** folder inside the **Windows Azure Storage for Android Sample App** project and double-click the **strings.xml** file.
     1. 
  2. Locate and click the **toolkit\_connection\_type** entry.
     1. 
  3. On the right pane, change the placeholder text by ‘**CloudReadySimple’** value.
     1. 
  4. Perform the same steps described above to provide the proxy service information:
     1. **cloud\_ready\_simple\_proxy\_service**: The full address where the proxy service is hosted *(e.g. https://yourproxyservice.cloudapp.net)*.

Configuring the application using the Proxy Services with Access Control

* 1. Navigate the **res/values** folder inside the **Windows Azure Storage for Android Sample App** project and double-click the **strings.xml** file.
     1. 
  2. Locate and click the **toolkit\_connection\_type** entry.
     1. 
  3. On the right pane, change the placeholder text by ‘**CloudReadyACS’** value.
     1. 
  4. Perform the same steps described above to provide the proxy service and Access Control configuration.
     1. **cloud\_ready\_simple\_acs\_namespace**: The Access Control namespace subdomain name. For instance if the namespace URL is https://wazmobiletoolkitdev.accesscontrol.windows.net you should provide **wazmobiletoolkitdev** as the value.
     2. **cloud\_ready\_simple\_acs\_realm:** The URI for which the security token that ACS issues is valid.
     3. **cloud\_ready\_simple\_acs\_symmetric\_key:** The 256-bit symmetric key for this namespace.
     4. **cloud\_ready\_simple\_acs\_proxy\_service:** The full address where the proxy service is hosted *(e.g.* [*https://yourproxyservice.cloudapp.net*](https://yourproxyservice.cloudapp.net)*)*.

Running the sample application

* 1. Open the **Android Virtual Device Manager** by navigating the **Window -> AVD Manager**.
     1. 
  2. Start one of the supported virtual machines (2.2 or above except the 2.3.3).
     1. 
     2. **Note**: If you have not installed any virtual machine, you can follow the steps provided as part of the [Installing and configuring the Android SDK](#ConfiguringAndroidSDK) section.
  3. Unlock the emulator.
     1. 
  4. Right-click on the **Windows Azure Storage for Android Sample App** project and select **Run As -> Android Application**.
     1. 
  5. If you configured the application to use a service proxy, you must login either providing your username & password or against one of the identity providers configured when using Access Control.
     1. 
     2. Username & Password Login Activity
     3. 
     4. Identity Provider selector Activity
  6. Once authenticated the storage service selection screen will appear as follows.
     1. 
     2. Storage Service Type Selector