Mobile Multimodal Interaction and Rendering Framework

Getting Started with the MMIG Starter Kit

Ehsan Gholamsaghaee, Aaron Ruß
05.06.2013

Table Of Contents

Getting Started

This chapter describes the system and software requirements for developing multimodal mobile interaction applications using MMIR-framework for Android operating system.

Supported Operating Systems

- Windows XP, Vista, or Windows 7
- Mac OS

Requirements

For development based on the MMIR Starter Kit application

- Download and install Eclipse (3.4+) (e.g. the *Classic*, or *EE Developer* edition)
- Download and install Android SDK (2.2+)
- Download and install ADT Plugin
- Download the MMIG-StarterKit.zip

You should also ensure that you have created at least one Android virtual device (AVD): in Eclipse select the **Java** perspective and open from the menu **Window ► Android Virtual Device Manager**; for detailed instructions on how to create a new AVD in Eclipse, we refer to the online instructions. You will need an AVD to run your project in the Android emulator. You should have the Android **tools/** and Android **platform-tool/** folders listed in your system path. Both these folders can be found inside your Android installation directory.

Use **Window ► Android SDK Manager** for installing the SDK of the Android version you are targeting with your application; at least one SDK has to be installed (we recommend to use at least ver. 2.2+).

Setting up Eclipse

While using Eclipse as development environment is not strictly required, we recommend it. The following gives some hints for setting up Eclipse in order to ease development of MMIR based applications.

First, you should download and install the Android SDK; you should note the directory where the SDK is installed into, as you may need this information later. Next, install the ADT Plugins (e.g. in Eclipse, using the update site); see also chapter Requirements (p. 2), above.

HTML Editing

Hint: If your Eclipse environment brings an HTML editor (e.g. as the **EE edition** does)[1] you can setup Eclipse to use this editor for eHTML files. eHTML is the template format used by the framework, similar to JSP (Java Server Pages) or ASP (Active Server Pages) templates.

For setting the HTML editor as default editor: **Preferences ► General ► Editors ► File Associations**, then in **File types**, add an entry for ***.ehtml** and set the HTML editor as its default.

---

[1] *Eclipse Classic* does not bring an editor HTML by default; you can install an editor manually e.g. using your Eclipse's update page (*Install New Software…*) *http://download.eclipse.org/releases/[version name]*, expand the entry *Web, XML and Java EE development* and select *Web Page Editor*.
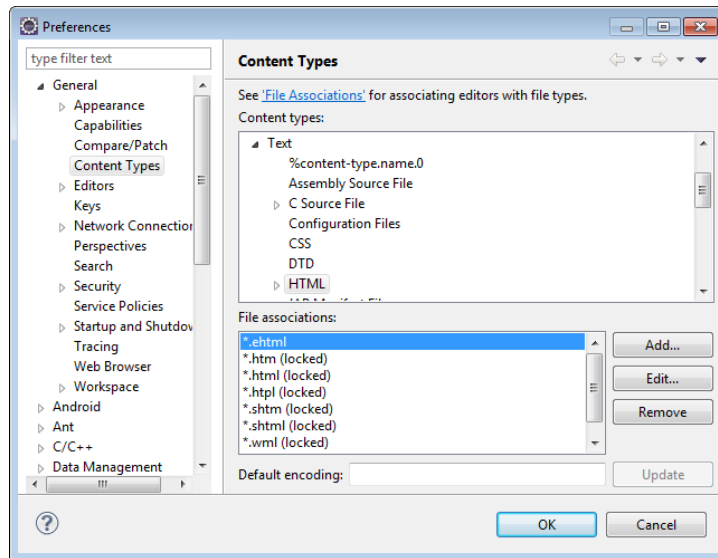
Figure 1: Eclipse Preferences dialog for associating file extensions with content types.

For using the syntax highlight of the HTML editor: in **Preferences ► General ► Content Types** select **Text ► HTML**, then in **Text ► File Associations** add an entry ***.ehtml**.

Figure 2: *.ehtml file without (left) and with (right) syntax highlighting for content type HTML in Eclipse.

Android Source Code

For debugging purposes, you may want to configure Eclipse to have access to the Android source code. For Android 4.x you can use the SDK Manager, e.g. in Eclipse in the Java perspective, open **Window ► Android SDK Manager**, expand the entry for the corresponding Android 4.x version and select **Sources for Android SDK** for installing the source. When prompted with No sources attached... for Android code (e.g. during debugging), select the `[Android SDK directory]/sources/[version]` subdirectory for the version currently used in your project from the Android SDK installation directory, e.g. `/android-sdk/sources/android-16`.

For some of the older Android versions[2], the plugin from http://code.google.com/p/adt-addons/ ►
Android Sources can be used to integrate source code support. You can use the update site **Fehler! Hyperlink-Referenz ungültig.** for installing the **Android Sources** plugin.

Importing MMIG-StarterKit into Eclipse

To import the MMIG-StarterKit (MSK) to Eclipse follow these steps:

1. Start Eclipse and go to **File ► Import.**
2. Select **Existing Project into Workspace** and click the **Next** button.
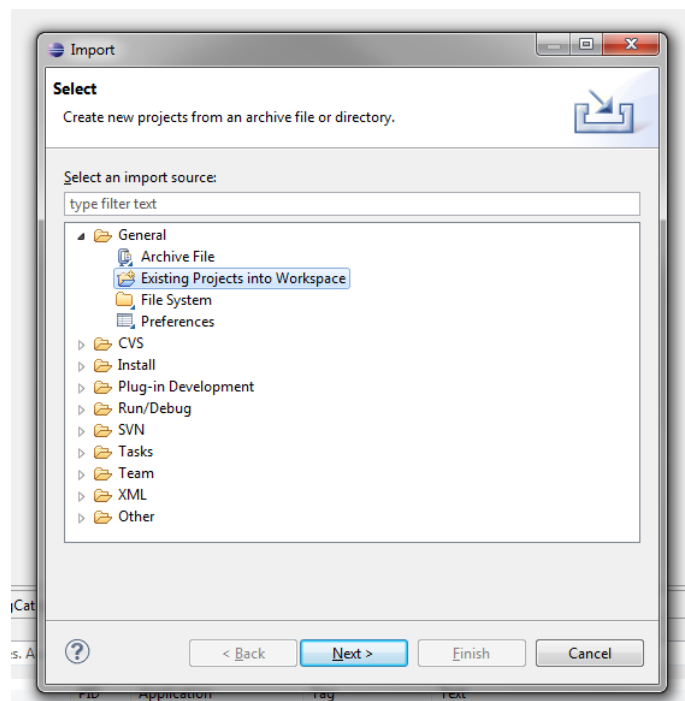
Figure 3: Import an Existing Project into Eclipse

3. Click the radio button next to **Select archive file** and click the **Browse** button on the following dialog.

---

[2] Note, that currently no source code is available for Android ver. 3.x.
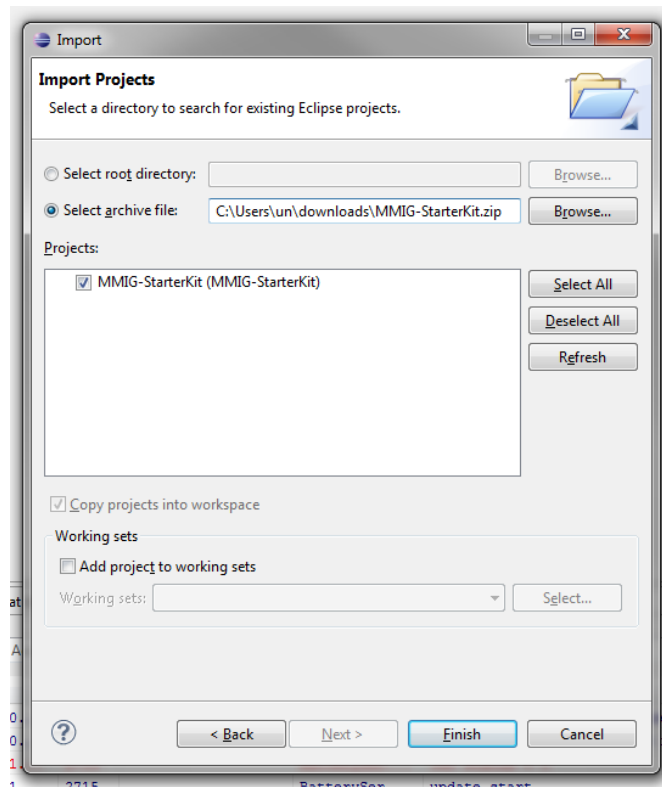
Figure 4: Select MMIG-StarterKit.zip

4. Navigate to MMIG-StarterKit.zip on your disk. Click **open** to select it.

MSK is based on the Apache Cordova platform and has almost the same structure as a Cordova project. Figure 5 illustrates the structure of MSK in Eclipse.
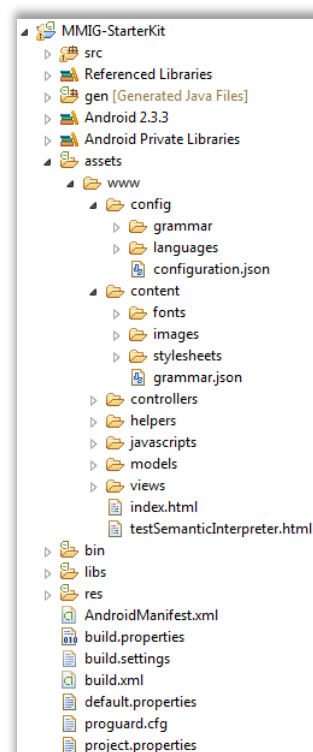
Figure 5: Structure of MSK

Common Problems

The following sections gives help for some common problem and issues that may occur when importing the MMIG-StarterKit into Eclipse.

If you experience compilation errors, some of the following steps may help:

- After importing, set the correct Android version: open the project's **Properties ► Android**, in the section **Project Build Target** select the Android version you target. If you reference other Android projects, ensure that they are correctly linked in the **Library** section; if in doubt, go through the steps to **Remove**, **Apply**, and re-**Add…** for the referenced projects.
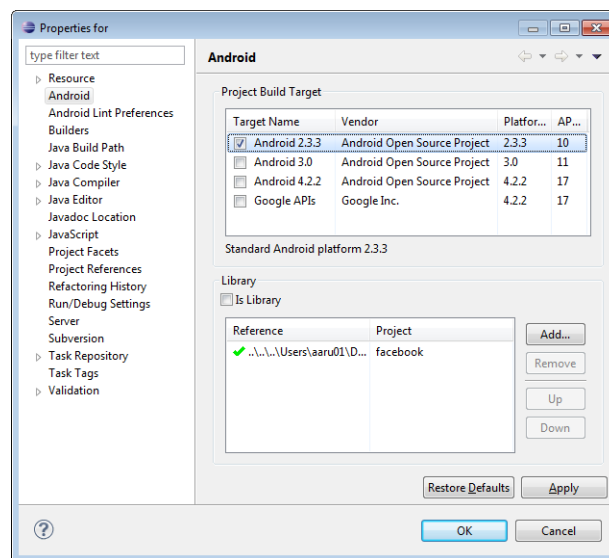

Figure 6: Selecting the Android version for the project

- Ensure that the correct Java compiler version is used: in the project's **Properties ► Java Compiler**, set the **Java compiler compliance level** to 1.5 (5) or 1.6 (6).
- For Android ADT version 22+: If you experience problems that classes cannot be found that are referenced by JAR libraries, go through the following steps:
  o ensure that all referenced libraries are located in the project folder `/libs` (or in a subfolder)
  o open the project's **Properties ► Java Build Path**
  o select the **Libraries** tab and ensure that all required JAR files are referenced (note that there is an entry **Android Private Libraries** which older ADT versions do not show)
  o Go to the **Order and Export** tab and check all boxes for the referenced JAR libraries (i.e. export them)
- If you remove or replace a JAR library in `/libs`, you should *"physically"* first remove the library, rebuild the project, and –in a case of replacement– then add the new library (and rebuild). Simply removing the library in the project's **Properties ► Java Build Path** may not be enough to work (in this case you may experience some error messages containing something similar to `Conversion to Dalvik format failed with error 1` in the **LogCat** output console).

Deploy to Simulator

- Right click the project and select **Run As** ►**Android Application**.
- Eclipse will ask you to select an appropriate AVD. If there does not exist one, then you need to create it.

Deploy to Device

- Make sure USB debugging is enabled on your device and plug it into your system (**Settings** ► **Applications** ► **Development**).
- Right click the project and select **Run As** ►**Android Application**.

Deployment for Web Site

- Open `assets/www/index.html` in a web browser *(see also Notes on compatibility below)*
- Alternatively, you can also host the contents of `assets/www/` on a web server

Notes on Browser **compatibility**:

- Google Chrome has additional restrictions for accessing local files. For Web Site Deployment as local file (i.e. directly opening /index.html in a browser) an additional command-line switch is required in order to allow the MMIR framework access to its local files:
  `--allow-file-access-from-files`
  Note, that
  - o This command-line option will only work, if no other instances of Chrome are running, when starting the application with this switch
  - o Afterwards, this command-line option applies to all (newly opened) instances of Chrome
  - o This option should not be used, when browsing the Internet, but only for development purposes, since it may pose a security risk

Notes on support for **speech** interactions:

- Currently, only Google Chrome (version 25.x.x) supports access to the microphone
  - o To enable Chrome for accessing the microphone activate the **Web Audio Input** in `chrome://flags/`
  - o On opening the `index.html` choose `Accept` for granting access to the microphone
- MSK is setup to use the Google web service for speech recognition. This requires converting the audio stream from the microphone in WAV format to the FLAC format. **TBD:** We plan to release a lightweight web service component in the near future (e.g. to be run on a Tomcat server), that allows conversion from WAV to FLAC