



# Transit Widget Emulator

## Getting Started Guide

Version: 1.2.2, build 20100817



# Table of Contents

1	Introduction.....	5
1.1	Project Information.....	5
2	Installation.....	6
2.1	Prerequisites.....	6
2.2	Recommendations.....	6
2.3	Installation.....	7
2.4	Un-installation and Updates.....	8
3	Using the Emulator.....	9
3.1	Basic Steps to Emulate a Widget.....	9
3.2	Optionally, You May.....	10
4	Managing Device Profiles.....	13
4.1	Device Profiles.....	13
4.2	Messaging Profiles.....	14
4.3	PIM Profiles.....	14
5	Advanced Features.....	15
5.1	Development Tips.....	15
5.2	Triggering Events.....	15
5.3	API Log Viewer.....	15
5.4	Import/Export Device Profiles.....	16
5.5	Multiple Virtual File Systems.....	16
5.6	Security Domain Emulation.....	17
5.7	JIL Packaging Specifications.....	17
5.8	JIL API Versions and Extensions.....	18
5.9	Directly Changing the SQLite Database (Risky).....	18
5.10	Development Practices to Avoid.....	18
6	Community Feedback, Contributions, and Support.....	20
6.1	Project Home Page.....	20
6.2	Support.....	20
7	API Notes.....	21
7.1	Notes as of Build 20100817.....	21

## Revision History

Date of Update	Document Version	Sections Changed	Description	Updates made by
<b>06 July 2010</b>	<b>1.2.2</b>	<b>All</b>	<b>New document</b>	<b>JIL</b>
<b>22 July 2010</b>	<b>1.2.2 build 20100722</b>	<b>All</b>	<b>Updated with latest information from build 20100722 and included some screen shots.</b>	<b>JIL</b>
<b>2 August 2010</b>	<b>1.2.2 build 20100729</b>	<b>All</b>	<b>Updated with information about API extensions and multiple JIL API support</b>	<b>JIL</b>

All rights reserved. © JIL B.V., the Netherlands - 2010

All intellectual property rights (including all copyrights, patents, trade marks and service marks) whether registered or unregistered, relating to this document and its content shall remain the property of JIL or its licensors. No licenses, express or implied, relating to the content of this document are granted. No part of this document may be reproduced or utilized, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of JIL or as expressly permitted by law.

This document is intended solely for general informational purposes only. Although care has been taken in drafting this document, this document is provided “as is” without any warranty of any kind, express or implied, including, but not limited to warranties regarding quality, accuracy, merchantability or fitness for a particular purpose. The use of this document is at the sole risk of the user and JIL assumes no liability or responsibility pertaining to the use of this document. In no event shall JIL be liable for any direct and/or indirect loss, damages or expenses whatsoever, suffered due or pursuant to the use of this document and/or the fact that this document is incorrect, incomplete or not up to date, except for loss or damages that have been caused by wilful intent or gross negligence of JIL.

This document constitutes confidential information and contains proprietary information belonging to JIL. This document is disclosed only to the recipient to whom this document is addressed and is pursuant to a relationship of confidentiality under which the recipient has obligations of confidentiality. The confidential information is to be used by the recipient only for the purpose for which this document is supplied. The recipient must obtain JIL’s written consent before the recipient or any other person acting on its behalf, communicate any information on the contents or subject matter of this document or part thereof to any third party. The third party to whom the communication is made includes an individual, firm or company or an employee or employees of such a firm or company. The recipient shall keep this document and any part thereof and any copies confidential and shall take all necessary measures to safeguard the confidentiality of this document or any part or any copy thereof to prevent any unauthorized disclosure, use, copying, publication or dissemination of the same to and/or by a third party.

This notice is without prejudice to any applicable rights and obligations agreed with JIL under any terms and conditions or at law.

JIL B.V.  
Schouwburgplein 30-34  
3012 CL Rotterdam  
THE NETHERLANDS  
[www.jil.org](http://www.jil.org)

# 1 Introduction

The Transit Widget Emulator is a community supported project sponsored by JIL as an alternative JIL widget emulator focusing on the following areas:

- Speed up JIL Widget development
- Provide useful features to all JIL developers
- Cross platform support
- Extensive device support
- Allow simulation of any possible use case
- Advanced widget debugging
- Past, current, and future standards support
- Develop a community project all JIL developers can contribute to and improve

Transit is not meant to replace the current JIL Widget SDK, rather supplement it. The JIL SDK features a complete virtual device based on the Android emulator which is useful for test cases utilizing actual device features such as receiving and placing calls, sending and receiving messages, file systems and peripheral hardware (camera, memory cards, etc.), as well as others. Transit excels early in the widget development life cycle where rapid emulation and multi-device simulation is most important. Transit aims to allow developers to manipulate and test every possible aspect of the JIL Widget API via manual triggering of events, populating API data, managing and persisting messaging and PIM data, advanced debugging, and more.

Transit is a community project in a beta/incubation phase that is very new. Some features are experimental or yet to be implemented, and may be buggy. We encourage developers to provide feedback, bug reports, and even code contributions at the projects home page noted at the end of this document. We hope the community will help us mold this into a project that we will all find invaluable.

## 1.1 Project Information

Home Page: <http://code.google.com/p/transit-widget-tools>

Current Stable Version: 1.2.2

Maintainer: Kyle Wilgus

## 2 Installation

### 2.1 Prerequisites

#### Firefox

Transit requires Firefox 3.6.3 or newer, this is a hard requirement as many required features are not available in older versions of the browser.

#### Operating System

In terms of operating systems, if Firefox can be installed, Transit supports it. Major platforms and architectures have been tested, however if you do run into any platform incompatibility issues, please submit a bug report on project home page (see the Community Contributions and Support section at the end of this document).

### 2.2 Recommendations

#### Display Resolution

Transit will run better on high resolution screens. As devices are packing more and more pixels into their displays, screen sizes are getting larger and larger. Emulating a device with a WVGA screen will take up the majority of a 1280x1024 display.

#### Firefox Profile

A separate Firefox profile is recommended for running Transit. Transit uses a SQLite database which can get bogged down if too many other extensions are installed and competing for resources. A typical profile would have Transit, Firebug, and Console2 installed. Including more extensions is possible, but should be done with caution.

#### Console<sup>2</sup>

Another Firefox extension which improves the usefulness of the Error Console (where JavaScript and CSS messages are displayed) is Console2. Widget developers will be spending a lot of time with the Error Console, and Console2 is highly recommended to make this time a bit easier. More information and downloads are located at the project's home page: <http://console2.mozdev.org>

#### Firebug

The Firebug JavaScript debugger can be used with the emulator to set break points, step through code, and examine objects in memory. To use Firebug when running the emulator,

launch the emulator with the "Emulate JIL Widget (debug)" sub-menu item. More information and downloads from Firebug are available at the project's home page <http://getfirebug.com>

### Security Precautions

Transit does allow widgets access to some protected browser resources which are normally not open to documents loaded from the local file system through extensions. Emulating only widgets from sources you trust is recommended, and loading un-trusted scripts and markup documents directly from the Internet should be avoided (as would be the case for most any widget). Adding a new profile is easy, use the following at your command prompt (same arguments for all operating systems):

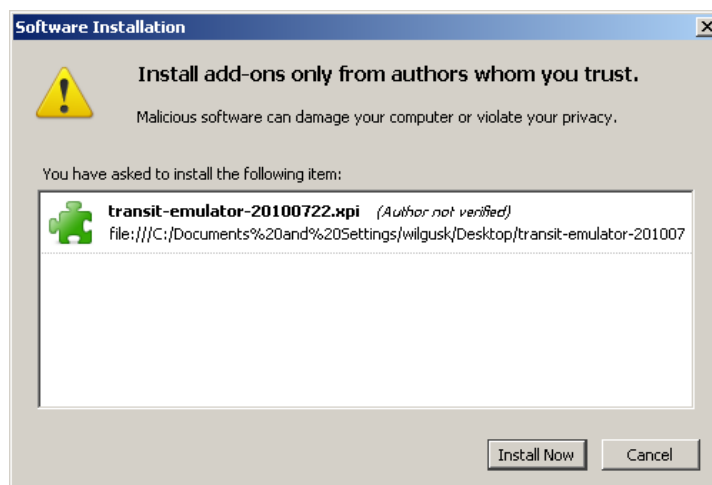
```
firefox -profilemanager
```

To launch Firefox with a specific profile, use the following command. Firefox will generally start with the last used profile if none was specified.

```
firefox -p profilename
```

## 2.3 Installation

Installing Transit is very simple. Transit is distributed as a Firefox extension package XPI file. Open your Firefox browser and create a new profile if desired. Open the File menu, choose "Open File..." and open the Transit XPI file. Firefox will pause for a few seconds and then allow you to install the extension by clicking the "Install Now" button as shown in the image below. A browser restart will be required before the extension will work.



*The add-on install dialog after opening the XPI file.*

The most recent development build as well as stable builds of the XPI file can be found at the project home page.

## 2.4 Un-installation and Updates

Transit can be uninstalled through the Firefox Add-on manager in the Tools menu. Transit will save your device, messaging, and PIM profiles, but it's always a good idea to back them up before uninstalling or deleting a profile. Removing the emulator will require a restart of Firefox.

To backup your profile database, copy the SQLite file in the "transit" directory within your Firefox profile directory. Save this file and use it to overwrite the new SQLite file once you have re-installed the Transit plug-in.

Typically, your profile directory is located in the following places (assuming the name of the profile is "default"). "XXXXXXXX" represents a random hash Firefox uses to create a unique profile directory.

**Windows:** C:\Documents and Settings\username\Application Data\Mozilla\Firefox\Profiles\XXXXXXXXXX.default

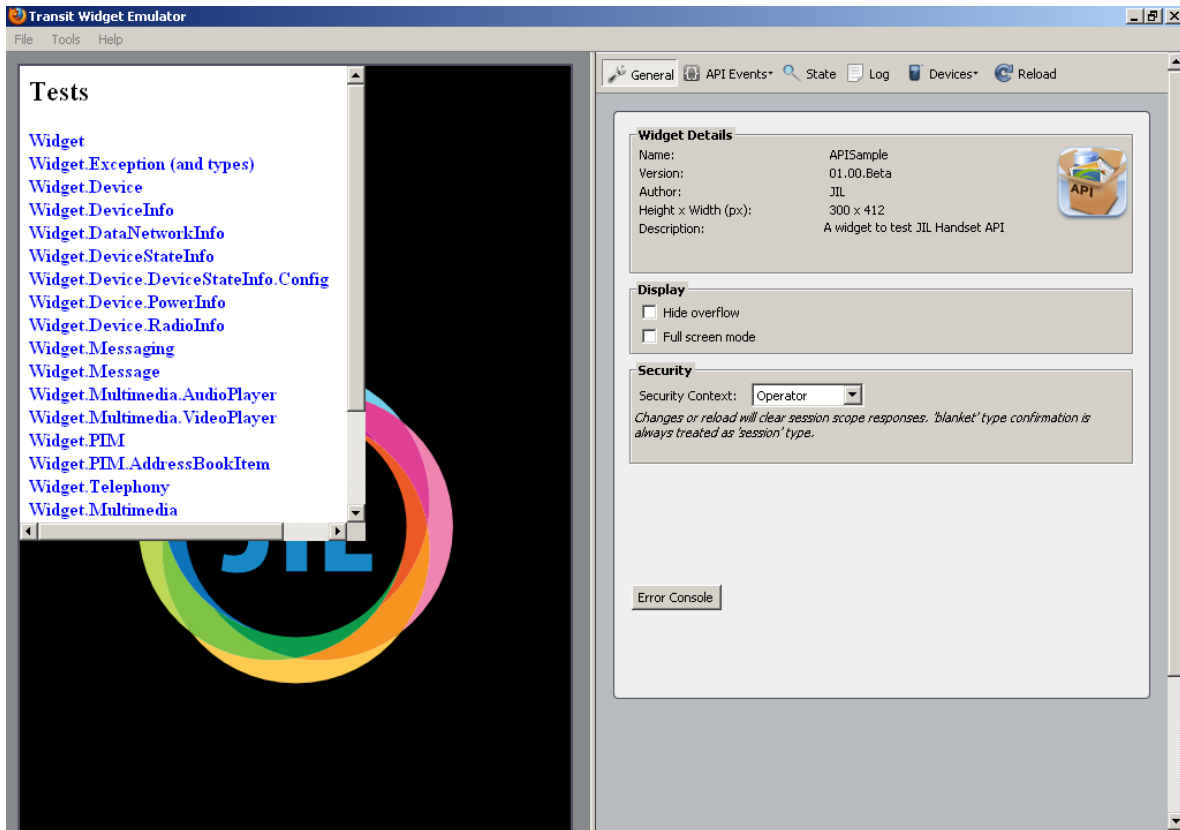
**Linux:** /home/username/.mozilla/firefox/XXXXXXXXXX.default

Development updates are posted on a weekly basis, usually on Thursdays. When Firefox opens up, it will generally check to see if there is a new version of the Transit plug-in available and allow you to install it. We recommend checking for updates at least once a week if you have this feature disabled or haven't restarted Firefox in a week.

Additionally, further updates to the Transit plug-in will automatically patch your database so there should be little worry of losing your profile data, but it's always a good thing to make backups. We intend to add a menu option to back up and restore via a button or two to make this process a bit easier.



menu. The widget will then be running in the emulator runtime environment immediately. Previously the emulator required a “Start” button to be pressed to initialize the runtime environment, but this is no longer needed and the “Start” button has been removed.



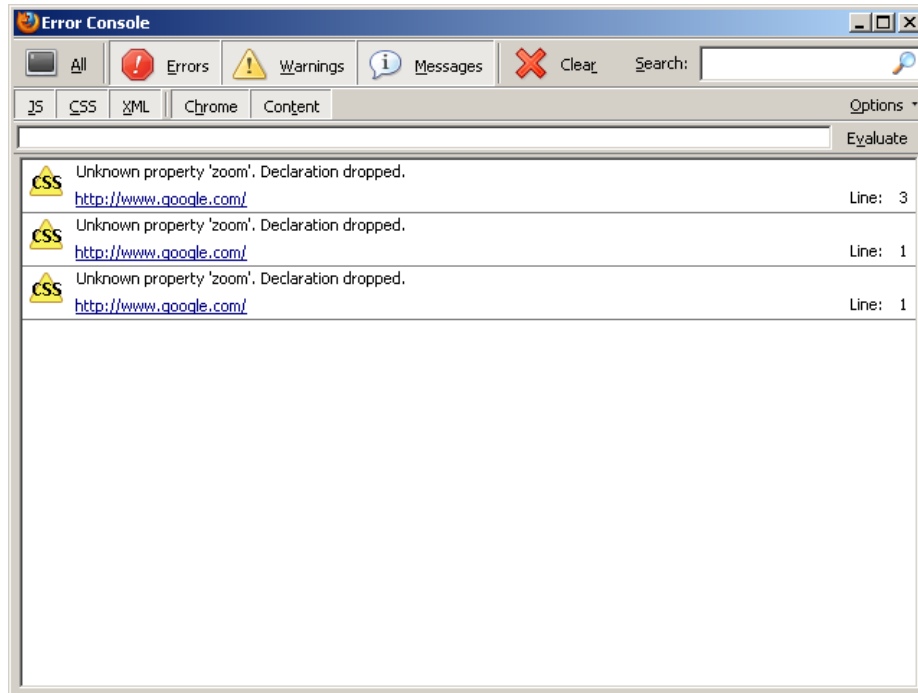
The Transit Emulator window with a test widget running.

5. Code changes can be made at any time in your widget editor while the emulator is running. Click the "Reload" button to refresh the widget and your changes will be reflected. Keep in mind a reload will reset the runtime context and profile information will be reloaded from the database as well.

## 3.2 Optionally, You May...

### Error Console

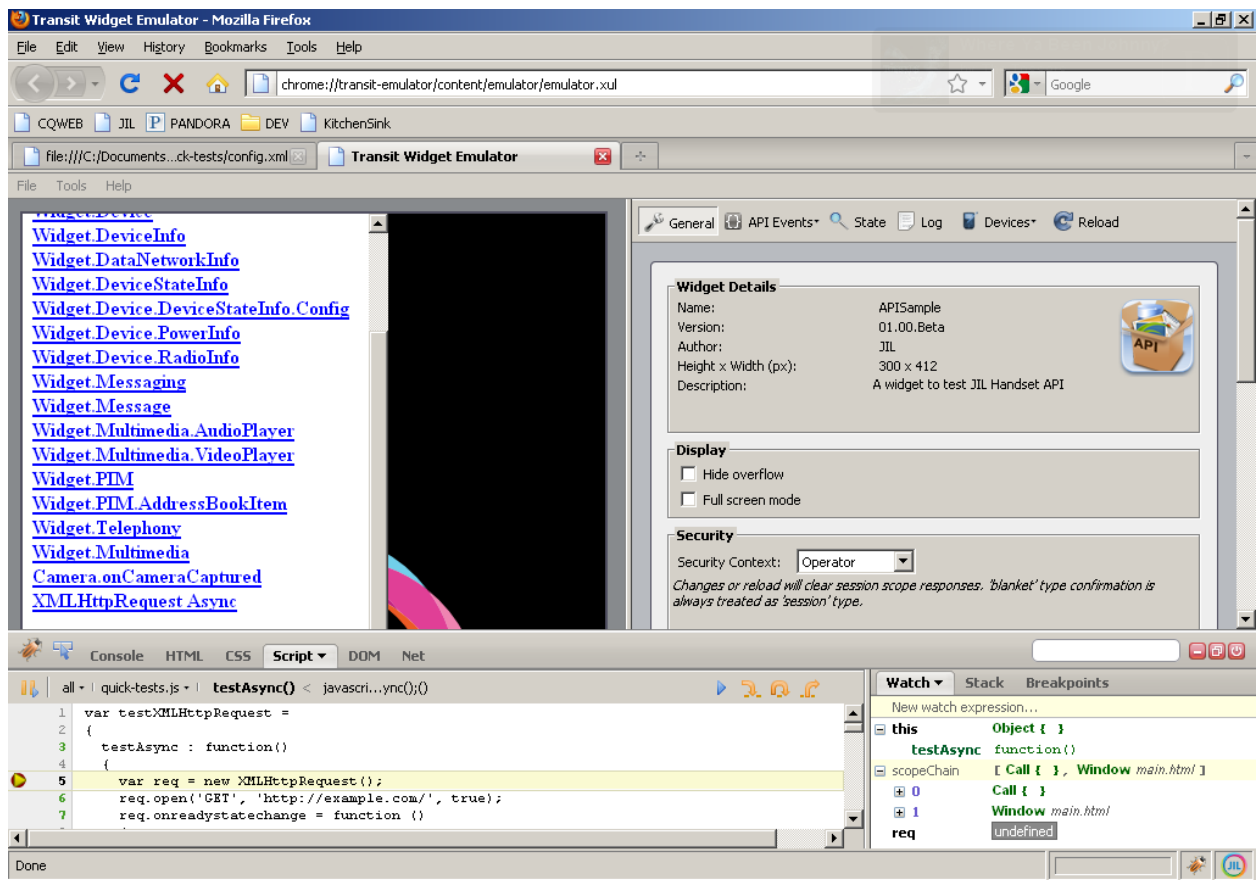
You may want to open the "Error Console" window (Tools > Error Console) to monitor any JavaScript or CSS errors/warnings produced by your widget or the emulator. For a more useful Error Console, install the Console2 extension as mentioned in the "Recommended" section above.



The Console<sup>2</sup> window.

### Firebug Debugging

You may want to run the widget with Firebug debugging enabled. To do so, use the "Emulate Widget (debug)" sub-menu item to load the emulator in a browser tab, allowing Firebug to be turned on. Firebug will allow you to debug Transit's code as well, but I recommend only debugging your widget's code as the internals of Transit are mostly loaded as XPCOM components are not visible with Firebug, which won't provide much useful information anyway.

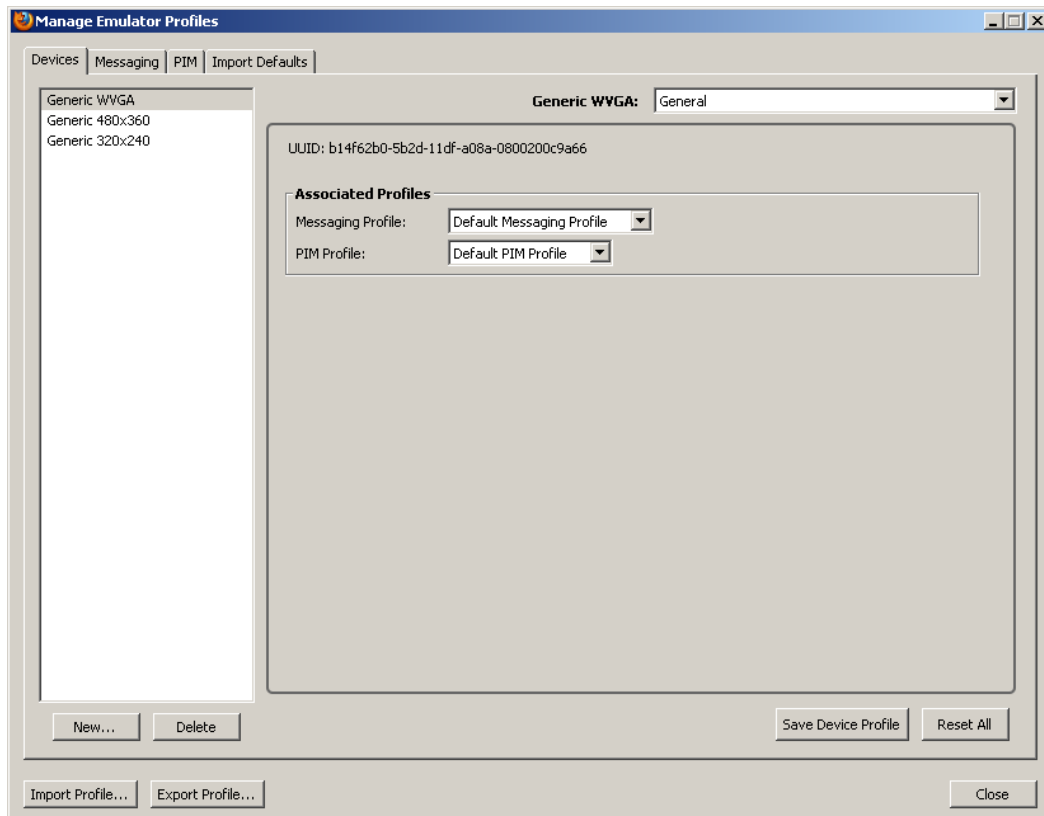


A test widget open in debug mode with Firebug showing a breakpoint set and ready to step through code.

## 4 Managing Device Profiles

Transit offers three types of profiles to use when emulating a widget and each can be managed individually through the "Manage Profiles" interface which is a sub-menu of "JIL Widget Emulator" in the "Tools" menu.

Messaging and PIM profiles are associated to a particular device profile, and it is device profiles that you will choose to use when emulating your widget through the emulator window. Profiles can be deleted; however there must always be at least one profile of each type.



The Manage Profiles dialog open showing a device profile "Generic WVGA."

### 4.1 Device Profiles

Device profiles contain all information describing a specific device (real or imaginary) such as screen size, widget runtime information, telephony, network, radio, and power information, etc. Device profiles are matched with a messaging and PIM profile that will be used when emulating under a particular profile.

## 4.2 Messaging Profiles

Messaging profiles contain all information used when handling the JIL Widget.Messaging API calls. Email accounts, folders, and messages are all managed here. Using multiple messaging profiles is useful when you have a diverse widget portfolio, wish to emulate different types of user behaviors, or different sets of test data. You can also export profiles to share with other developers so they can work off the same data sets.

## 4.3 PIM Profiles

PIM profiles contain all information used when handling the JIL Widget.PIM API calls. Address book items, calendar events, address book groups and address book attributes are all managed here. Using multiple PIM profiles is useful when you have a diverse widget portfolio, wish to emulate different types of user behaviors, or different sets of test data. You can also export profiles to share with other developers so they can work off the same data sets.

## 5 Advanced Features

### 5.1 Development Tips

#### Detecting Emulator Container

It is possible to detect when your widget is running within the Transit Emulator. This may prove useful for testing specific use cases, or any other purpose. To do so, you can check the following flag in your widget's JavaScript code:

```
Widget.Device.isEmulator
```

Note that this flag is not part of the JIL API spec and simply is provided as a convenience.

### 5.2 Triggering Events

All events that are not triggered directly by an API (find functions, etc.) call can be manually triggered via the "API Events" tab in the emulator window. The events are organized by API in the drop down menu. Click the radio button next to the event you would like to trigger, and then event context information will be displayed. You can change any context information that may be present in the API when an event is triggered. As the emulator says, contextual information entered here will NOT alter the device profile. Events that are triggered by API searches (asynchronous find\* methods) can only be triggered via an API call, not the event interface.

Additionally a delay in milliseconds can be added before the callback functions are called. This is useful for simulating a real world delay that might be encountered on an actual device for calls like requesting location, searching messages or files, and more.

IF you would like to see what the source code of the function that is set for an event callback, developers can now click the "Callback Source" link near the Trigger button to see what function is set for the currently selected event context.

### 5.3 API Log Viewer

One of the original aims of Transit was to provide a record of each API call and useful contextual information for developers to review after executing a use case or test to make sure their code did what was expected. In the emulator window, there is a "Log" tab containing a text box with each log entry; the most recent entry is listed first, each with a time stamp. The log can be cleared manually by clicking the "Clear" button, and is cleared automatically when the widget is reloaded. The logs aren't persisted, be sure to copy/paste interesting log entries to a buffer or text file.

## 5.4 Import/Export Device Profiles

Device, messaging, and PIM profiles can be imported and exported in the "Manage Profiles" interface. This is useful for downloading future device profiles released by JIL or a 3rd party; even sharing profiles with other developers and for submitting bug reports. When a device profile is imported, it will automatically inherit the oldest available messaging and PIM profiles. This can be changed after import through the "General Information" pane in the device profile. Additionally, some data is device agnostic and default values will be used. These default values can be changed through the "Import Defaults" tab on the "Manage Profiles" interface if you prefer other values. Profiles are simply stored as JSON notation in a file, it's possible to manually edit these, but definitely not recommended as a typo can easily break them.

## 5.5 Multiple Virtual File Systems

Transit provides the ability to add multiple file systems to a device profile, essentially emulating a device that handles peripherals (like memory cards) as separate mount points. In the device profile, under the "Widget.Device" pane, adding and mapping virtual file systems is straight forward. As stated in the JIL API spec, JIL represents a file system similar to UNIX, using "/" forward slash as a directory separator. When referencing a file in your widget's code, Transit will handle mapping the virtual file system to the local file system for you, accessing files already on your locally mounted partition.

Here is an example of a device with three virtual file systems:

Device File System Root	Mapped to Local Directory
/sdcard	/home/developer/projects/jil/widgets/data/sdcard
/app	/home/developer/projects/jil/widgets/data/app
/app/photos	/home/developer/projects/jil/widgets/sample/photos

Example file mappings using these three virtual file systems:

Device File System Path (in your widget's code)	Mapped to Local Path
/sdcard/mywidget/file.txt	/home/developer/projects/jil/widgets/data/sdcard/mywidget/file.txt

/app/file.txt	/home/developer/projects/jil/widgets/data/app/file.txt
/app/photos/mytrip/picture.jpg	/home/developer/projects/jil/widgets/sample/photos/mytrip/picture.jpg

## 5.6 Security Domain Emulation

Transit implements the JIL security domain requirements per the JIL specification. A widget can be run under three possible domains:

**unidentified:** an unsigned widget will be automatically run in the unidentified security domain. Some protected APIs are disallowed and most will prompt the user to approve access to the resource by the widget.

**identified:** a widget signed by the JIL authority, but not the individual operator authority. This is the most common domain.

**operator:** a widget signed by the local operator authority. This domain grants the most privileges to the currently running widget.

Security domains are called contexts and by default, Transit will run a widget in Operator mode. To change the security context for your widget on the fly, there is a drop down menu on the General Settings tab in the emulator window. Reloading the widget will reset the session responses, and the user will be asked to confirm access as if it was the first time they have run the widget.

Also note that the "blanket" confirmation type (confirmation is asked once and never again for the user) is treated the same as the "session" type to be of more use for developers. Transit will never treat a security confirmation as permanent.

## 5.7 JIL Packaging Specifications

Transit currently supports the JIL 1.0 and 1.2 packaging specifications. Which packaging specification a widget uses has little to do with how it will function as a widget, but is important for ensuring device compatibility as many JIL devices will only support one version or the other.

When your widget is loaded, Transit will determine the packaging specification that is used and display that information in the general tab of the emulator screen. If the config.xml file is missing some elements, it may still be able to emulate the widget using "Unknown" as a temporary values (these will not be saved), however other values are required to emulate

and must be present to emulate (widget Id, version, author, etc.). In both cases, the developer will be warned that there is missing information (required or not), allowing the developer a chance to fix the missing pieces and load in the emulator again for full support.

## 5.8 JIL API Versions and Extensions

The Transit Emulator has recently introduced support for multiple versions of the JIL API. Currently versions 1.2.2 and 1.1 revision 4 are supported. Device profiles can be configured to support one or the other via the Manage Profiles window. This is a mandatory field, and currently 1.2.2 is the default setting for new device profiles. We are currently working on adding support for the next JIL API specs (1.3 and beyond) in hopes to release the API specification and emulator support at the same time.

Another API feature recently added is the use of “API Extensions” which can be enabled for a device profile. API Extensions are meant to supplement JIL APIs that may be present on one device, but not another, or 3<sup>rd</sup> party APIs (like BONDI for example). Currently, only one extension is available, which is the “Samsung M1/H1” extension which adds the “widget.getURL()” function with the lowercase “w.” Enabling this extension on a device profile will correctly emulate the available objects on the Samsung M1 and H1.

We’re always looking for more extensions to add to Transit, if you are aware of APIs outside of the JIL API spec that a Widget Runtime is using, let us know at the project home page, and we’ll work to bring those APIs in as extensions quickly.

## 5.9 Directly Changing the SQLite Database (Risky)

It's not supported or recommended, but device profile information is stored in a SQLite file in the Firefox profile's folder in the “transit” folder. This file can also be deleted and Transit will use the default SQLite file next time the emulator is launched (only do this if you don't mind losing all of your profiles or have everything backed up).

If you are comfortable with SQLite, have an editor, and don't mind risking the corruption of your profiles, these files can be changed directly if for some reason the "Manage Profiles" interface is not adequate.

## 5.10 Development Practices to Avoid

When using the Transit emulator, there are practices that should be avoided if possible:

## JavaScript Prompts

Refrain from using built in JavaScript prompts such as alert, confirm, etc. as these are modal and will freeze the emulator window until the dialog is cleared. This will keep you from being able to trigger events, view the emulator log, reload profiles, etc. while the prompt has taken control of the window.

## 6 Community Feedback, Contributions, and Support

### 6.1 Project Home Page

Details on known issues, current activities, feature support, road map, etc. can be found at the project's home page (<http://code.google.com/p/transit-widget-tools>). Bug reports and source code can also be found here.

### 6.2 Support

Support for Transit is currently provided through the community process. We encourage discussion with other JIL developers or project maintainers to address any issues you might be having. Bug reports and patches are greatly appreciated when these issues do pop-up as well.

## 7 API Notes

### 7.1 Notes as of Build 20100817

The following JIL APIs currently have limitations in their implementation for the JIL 1.1r4 and 1.2.2 Widget API Specifications. The intention is to address these limitations quickly in upcoming releases.

JIL API	Limitation or Note
Widget.PIM.onVCardExportingFinish	Not yet implemented.
Widget.PIM.AddressBookItem.setAttributeValue	Not setting attributes that are defined as fields of the AddressBookItem object. The requirement is not clear in the JIL specification whether this should work or not. Currently requesting clarification of the requirement from JIL.
Widget.Device.File.createDate	File create date is not available through the Mozilla nsIFile interface, currently setting the value to the same as the file's lastModifiedDate field value.
Widget.Multimedia.Camera.captureImage	Camera integration has been implemented yet. Currently working to integrate a connected webcam if available.
Widget.Multimedia.Camera.startVideoCapture	Camera integration has been implemented yet. Currently working to integrate a connected webcam if available.
Widget.Multimedia.Camera.stopVideoCapture	Camera integration has been implemented yet. Currently working to integrate a connected webcam if available.
Widget.PIM.exportAsVCard	Not yet implemented.
Widget.PIM.onCalendarItemAlert	Calendar alert events are not triggered by an actual clock. Alarms must be triggered manually through the "API Events" interface.
Widget.Messaging.Message.saveAttachment	Currently works as a copy operation, taking the file referenced in the attachment and copying it to the specified file location.
Widget.Messaging.createFolder	When creating a folder, the folder defaults to "INBOX" folder type.
Widget.Messaging.getMessage	The "index" field is treated as the position of the message in the folder sorted by message date, descending.
Widget.Multimedia.Camera.setWindow	Camera integration has been implemented yet. Currently working to integrate a connected webcam if available.

JIL API Security restrictions are applied to widgets with the Transit emulator, see the section Advanced Features in this document. Currently only functions are being monitored by the security manager and not API properties (Widget.Device.AccountInfo.phoneMSISDN, etc).