Developer Manual

Introduction

This manual is aimed at people who feel obliged to continue the development of Virtual Controller.

Dependencies

- Java 6 SE development environment
- Android SDK
- An Android device

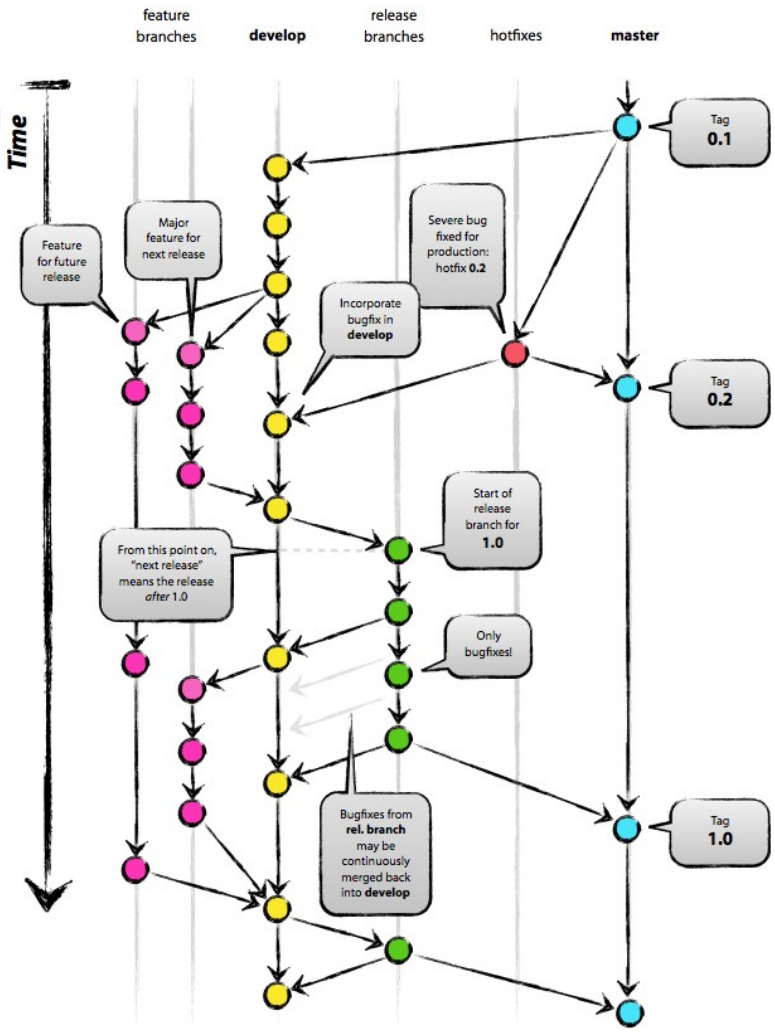
Android SDK targets

- Minimum SDK: 15
- Target SDK: 18

Getting started

To get started working on this project, install Git and create a user account on https://github.com/. We are working as closely as possible to the branching model below.

Checkout and pull developer branch, and start Eclipse. Import VirtualGamepadHost and VirtualGamepadLib. Add the bluecove jars as libraries and the VirtualGamepadLib project in the VirtualGamepadHost buildpath. Import VirtualGamepad as “existing android code into workspace”.



To implement a new feature, create a new branch according to the branching model

-

Building and installing

Building

Build VirtualGamepad.apk using Eclipse:

Code
Project -> Build Project

Create .jar file for the VirtualGamepadHost using Eclipse:

Code
File -> Export Java -> Runnable JAR file Choose ‘Package required libraries into generated JAR’

Commit changes that you have done to the git repository:

Code

$ cd <path-to-local-git-repository>
$ git add <file/s to add>
$ git commit -m “Insert message”
$ git push origin <branch>

Installing

To install VirtualGamepad.apk in a started emulator:

Code
$ cd <Android SDK/tools/> $ adb install <path-to-bin/VirtualGamepad.apk>

To install VirtualGamepad.apk in a connected Android device:

Code
$ cd <Android SDK/platform-tools/> $ adb -d <path-to-bin/VirtualGamepad.apk>

To run VirtualGamepadHost.jar:

Code
$ java -jar <path-to-jar/VirtualGamepadHost.jar>

Release procedure

Licensing

We have used the GPL 3.0 license. You must therefore follow the procedure as described on http://www.gnu.org/copyleft/gpl.html, and include the text snippet at the top of all of the source files that we have edited.

Release notes

The release notes should contain the following:

- New features
- Updated features

Releasing

After everything has been performed accordingly, push to the master branch. You must also add the release notes to the readme file in the repository.

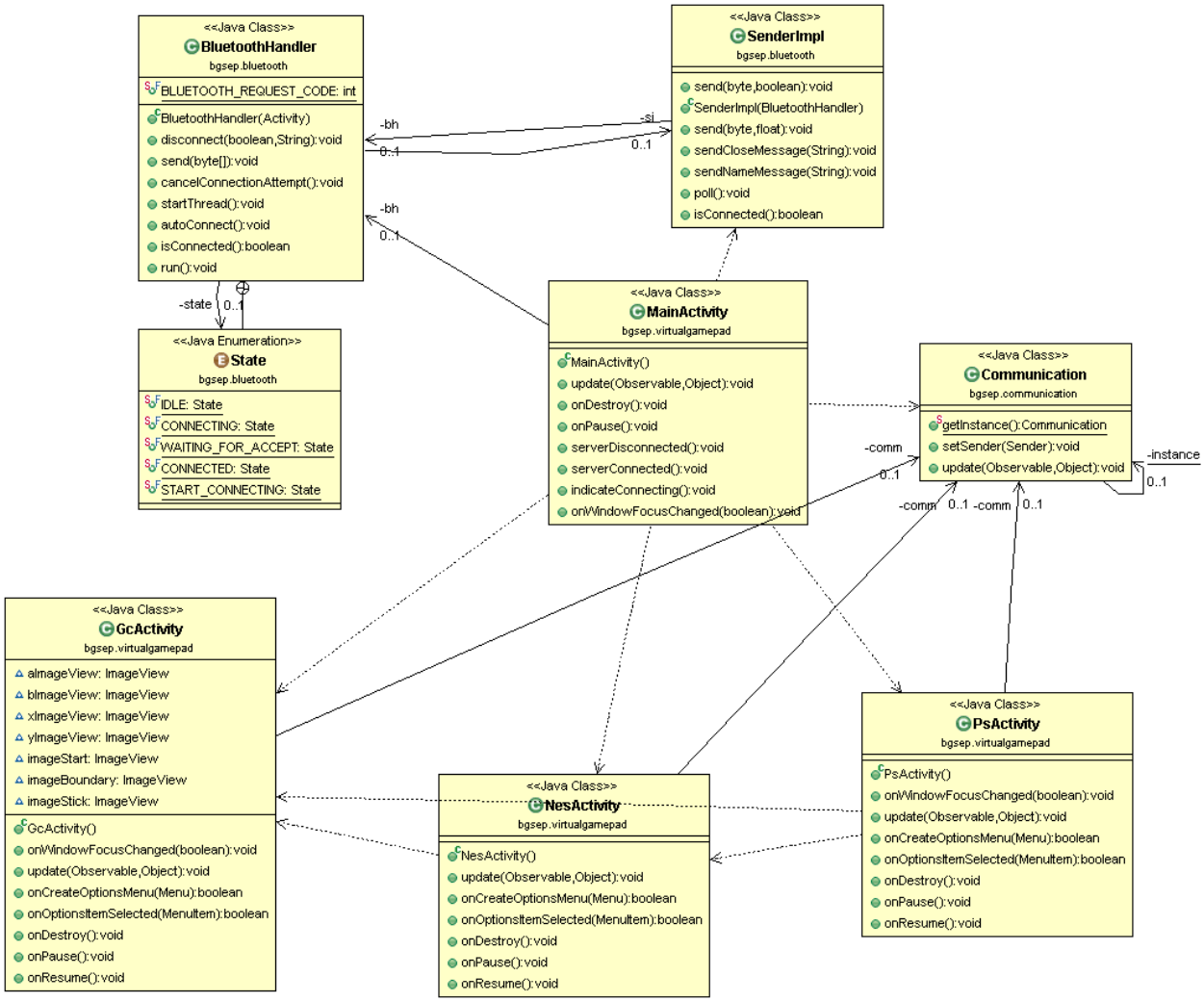
Architecture

The android application

The application uses the MVC model to some extent. The code is divided in packages in the following categories:

- **communication** - General server communication.
- **model** - Handling of input generated from the user. Responsible for updating the views and the Communication part.
- **virtualgamepad** - All Activities goes here. This is the View part of the MVC.
- **bluetooth** - Communication with the server using Bluetooth RFCOMM.
- **test.bluetooth** - Bluetooth specific tests.

Any additional communication techniques should implement the Sender interface in the communication package.



UML diagram of some important classes in the android application.

The server side application

The VirtualGamepadHost is the server side application that mainly starts up an RFCOMM socket and handles clients.

It has a multithreading architecture with one thread that listens to new connections and starts up client threads. Every client that is connected to the server is its own thread containing an input and output stream.

The BluetoothClient class, that represents the clients, uses ‘java.awt.Robot’ to convert the incoming data to keyboard presses.

UML diagram of some important classes in the server side application.

Packages:

- **bluetooth** - Everything is related to the bluetooth connection is located in this package.
- **util** - The utilities to the server (i.e. IDHandler) is located here.
- **host** - Everything else that doesn't fit in the above packages is located here.

Testing

Testing is a crucial part of the developing process. Try to do function tests on all possible test combinations for each added feature. JUnit is used on the most critical functions. An overall shake test of the whole application is done before every release. It is important to test that nothing has been broken in the merging of branches. When fixing a bug, write in the commit message what problem that has been fixed.