# Deployment instructions

Follow these steps in order to integrate FUV with your application.  
You can either use the prepared binaries or checkout the project and compile or integrate into your own code.

## general pre-requisites

### Anti-Virus installation

If you want to use the AntiVirus module, you have to install one. We tested the application with ClamAV (open source).

* Download the latest version [from here](http://www.clamav.net) (currently 0.97).
* Install on your server. ClamAV can be installed globally as root or locally to the user without privileged account. You may install it from the user that runs the system or set the appropriate permissions and install from any login (we don’t require root installation).  
  [Follow this page for short installation instructions](http://www.clamav.net/doc/latest/html/node13.html). Notice that if you install not from root, you have to use the –disable-clamav flag.
* Set CLAMAV\_PATH as an environment variable for ClamAV path (required by the ‘av\_wrapper.sh’ script.  
  For example : export CLAMAV\_PATH=/home/users/myuser/clamav-0.97/bin/clamscan

### application Configuration file

Create a configuration file (file-validator-config.xml) for your specific application (or modify the example file). The directory containing the file should be in the application’s classpath (accessible by ClassLoader.getResourceAsStream()).

### logback configuration

FUV uses logback. Please refer to [Logback’s manual](http://logback.qos.ch/manual/configuration.html) for configuration instructions. If you want to override the default configuration (and you probably want to set log directory, rollback strategy,…) you’ll have to place in your classpath the ‘logback.xml’ and set it to match your application’s settings.

## checkout the code

FUV code is hosted by Google code. You can find the project’s page [here](ttp://code.google.com/p/fuv).  
For direct checkout (SVN) use: *svn checkout http://fuv.googlecode.com/svn/trunk*

Create the project’s JAR file by : *mvn package*.

## use the binaries

Alternatively, you can use the provided binaries.   
At the ‘binaries’ directory you’ll find two JAR files:

* The project itself – Contains only the project’s code and should be used with all required dependencies as specified in the POM file.
* Project + dependencies – Contains all required dependencies and therefore can be used as is. Much bigger than the previous file and at most cases contains dependencies that you already have in your application. So be wise with using it…