**Asterisk-Java**

The [Asterisk-Java](http://asterisk-java.org/) package consists of a set of Java classes that allow you to easily build Java applications that interact with an Asterisk PBX Server. [Asterisk-Java](http://asterisk-java.org/) supports both interfaces that Asterisk provides for this scenario: The [FastAGI](http://www.voip-info.org/wiki/view/Asterisk+FastAGI" \o "Asterisk FastAGI) protocol and the [Manager API](http://www.voip-info.org/wiki/view/Asterisk+manager+API).  
  
The FastAGI implementation supports all [commands](http://asterisk-java.org/development/apidocs/org/asteriskjava/fastagi/command/package-summary.html) currently available from Asterisk.  
  
The Manager API implementation supports receiving events from the Asterisk server (e.g. call progess, registered peers, channel state) and sending actions to Asterisk (e.g. originate call, agent login/logoff, start/stop voice recording).

# Asterisk FastAGI

# FastAGI

Implements the Asterisk Gateway Interface ([AGI](http://www.voip-info.org/wiki/view/Asterisk+AGI)) over TCP sockets. This can help alleviate CPU load on your telephony server by relocating resource hungry scripts to another networked server. In order to instruct Asterisk to attempt a network connection, you must supply the hostname or IP address of the server where your FastAGI service is hosted and preface it with agi://:

exten => 5551212,1,AGI(agi://192.168.0.2)  
  
Note that the agi:// syntax can be used with both the [AGI](http://www.voip-info.org/wiki/view/Asterisk+cmd+AGI) and [DeadAGI](http://www.voip-info.org/wiki/view/Asterisk+cmd+DeadAGI" \o "Asterisk cmd DeadAGI) dialplan commands, while [EAGI](http://www.voip-info.org/wiki/view/Asterisk+cmd+AGI) is currently not supported. The previous example instructs Asterisk to make a network connection to a host running on a local network server when the 5551212 extension is matched. By default, Asterisk attempts to connect to the specified server over port 4573. The port can also be specified should you choose to host the AGI service on another port:  
  
exten => 5551212,1,AGI(agi://192.168.0.2:8675)  
  
A request may also be specified in the FastAGI call, so that you can host multiple AGIs in a single network location. This request allows the FastAGI service to differentiate between different locations in the dialplan. For example:  
  
exten => 5551212,1,AGI(agi://192.168.0.2/GetCallerRecord)  
  
and:  
  
exten => 5551212,1,AGI(agi://192.168.0.2/CallerWantsCustomerService)  
  
The request portion of these calls will be received by the FastAGI service as the **agi\_network\_script** AGI variable. Asterisk will also include the **agi\_network** AGI variable. For example, the FastAGI in the previous example would receive something like this when called by Asterisk:  
  
agi\_network: yes  
agi\_network\_script: CallerWantsCustomerService

## Passing Arguments to FastAGI

### Asterisk 1.6.x

In Asterisk 1.6.x, arguments passed to the AGI command are passed through to the FastAGI as AGI variables. For example, calling a FastAGI with the following:  
  
exten => 5551212,1,AGI(agi://192.168.0.2/CallerWantsCustomerService,${EXTEN},${UNIQUEID},${CALLERID(name)})  
  
will result in the following AGI variables being passed to the FastAGI service:  
  
agi\_network: yes  
agi\_network\_script: CallerWantsCustomerService  
agi\_arg\_1: 5551212  
agi\_arg\_2: 1171048538.5875  
agi\_arg\_3: Mark Spencer  
  
The method described below for both Asterisk 1.2 and 1.4 is also still supported.

### Asterisk 1.2 & 1.4

With Asterisk 1.2 through 1.4, when using standard AGI, you can pass variables on the command line to your local scripts. While this is not possible with FastAGI, it can be simulated using **agi\_network\_script** and a query syntax similar to that used by HTTP. For example:  
  
exten => 5551212,1,AGI(agi://192.168.0.2/GetCallerRecord?extension=${EXTEN})  
  
In this case, the following AGI variables will be passed to the FastAGI service:  
  
agi\_network: yes  
agi\_network\_script: /GetCallerRecord?extension=5551212  
  
It is the responsibility of your FastAGI framework to parse this information out of the **agi\_network\_script** variable. Some FastAGI implementations already do this for you, so be sure to check the documentation for your particular framework.