NETGEAR

Smart Network

Developers Guide

350 East Plumeria Drive
San Jose, CA 95134
USA

January 2012
202-10970-01
v1.0

Technical Support

Thank you for choosing NETGEAR. To register your product, get the latest product updates, get support online, or for more information about the topics covered in this manual, visit the Support website at *http://support.netgear.com*.

Phone (US & Canada only): 1-888-NETGEAR

Phone (Other Countries): Check the list of phone numbers at

http://support.netgear.com/app/answers/detail/a_id/984.

Trademarks

Statement of Conditions

To improve internal design, operational function, and/or reliability, NETGEAR reserves the right to changes to the products described in this document without notice. NETGEAR does not assume any liability that may occur due to the use, or application of, the product(s) or circuit layout(s) described herein.

Contents

Chapter 4 Basic FCML

Chapter 5 Router Management APIs

Chapter 6 Control Point APIs

Chapter 7 Portal Client UI APIs

Appendix A Region, Country, and ISP Codes

Index

Introduction

1

A distributed software platform

Smart Network is an intelligent network that makes home devices easier to connect and use. The customer connects only the devices that are needed and customizes what those devices do and the type of traffic they run. Customers can access their system from an internet-connected personal device to configure, monitor, and view the system from wherever they are. The platform has a development environment and an SDK for developers to either write their own client plug-ins or system-level classes to extend Smart Network.

This chapter presents an overview of Smart Network:

- *Platform Architecture*
- *FCML Communications*
- *Plug-In Client Applications*
- *NETGEAR Control Point*
- *Control Point Architecture*

For more information about the topics that are covered in this manual, visit the support website at *support.netgear.com*.

Platform Architecture

Smart Network is made up of the components that are described here and includes the FLUID Control Markup Language (FCML). FCML is a language similar to XML that handles communications between the components. See *FCML Communications* on page 12.

User Interface device. A personal device such as a computer, touch panel, TV, smartphone, or handheld remote. A UI device enables secure access to the web-based user interfaces that run on the portal server and on control points. Customers and administrators log in to these web pages to view, administer, and monitor the configuration for their installation.

Portal server. A server hosted in a remote location that runs the Smart Network application server. The application server manages updates and changes to a configuration. The portal server communicates with smart devices through a control point.

control point. A broadband device at the installation site such as a gateway, router, network-attached storage (NAS) device, or set-top box that has a NETGEAR Control Point installed. The control point is the communication point between the portal server and the smart devices running in the home network.

NETGEAR Control Point is a Java-based platform that manages the software services that run on the control point. The NETGEAR Control Point development environment and SDK let developers enhance, create, and deploy client plug-ins that run on the control point or on smart devices.

Smart devices. Home or small–medium business devices that connect to the network and that NETGEAR Control Point controls. Examples are gaming consoles, TVs, and set-top boxes. Smart devices load the configurations that are stored on the portal server.

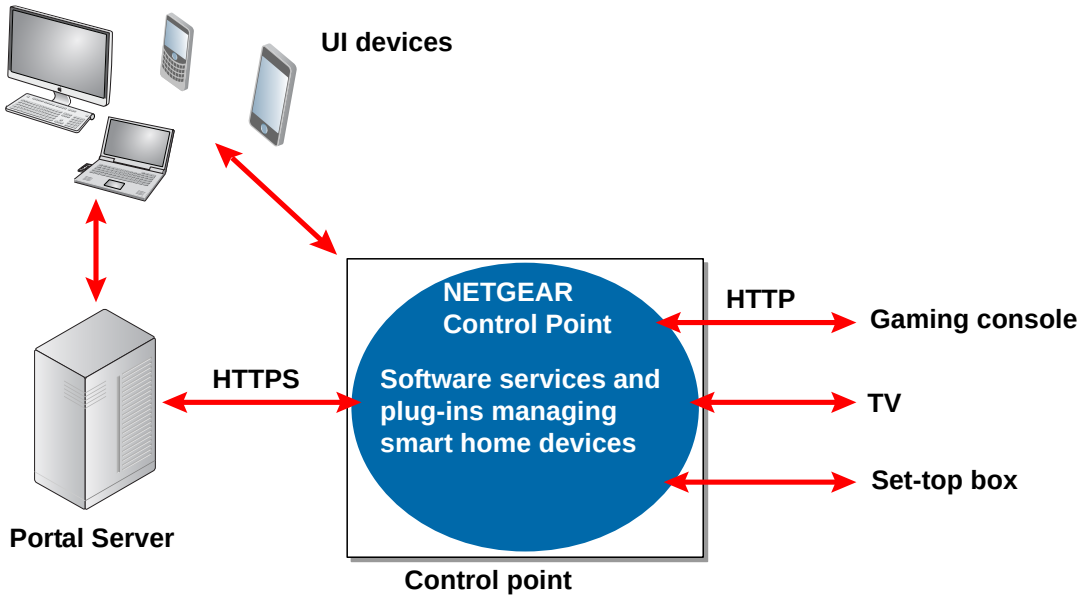

Figure 1. Smart Network components

FCML Communications

Smart Network provides FCML to handle all asynchronous request and response communications. Developers use FCML request and response messages in client plug-ins to manage network communications between the client plug-in and other components that run in Smart Network.

In the figure, the blue lines represent the client (Portal UI or control point) requesting messages, while the red lines indicate the sending of messages. These two operations are done asynchronously and in parallel.

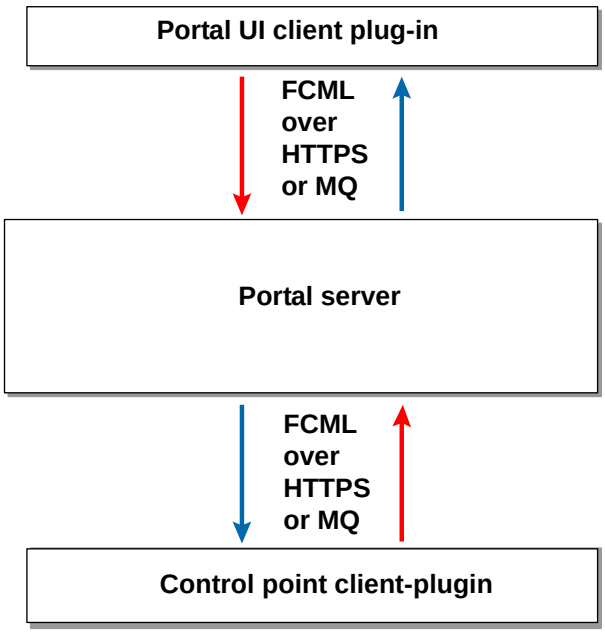

Figure 2. Flow from the UI through the portal to the control point and back

FCML has an open grammar, is program-language agnostic, and anyone can create a client plug-in that sends and receives FCML messages. There is a security layer that requires a client plug-in to be signed before it can be installed on and run in a NETGEAR Control Point.

Any of the following languages work with FCML and are suitable for writing remote client plug-ins: Flash, Java, AJAX, OCAP, C++, Objective C, and many more. Local services and client plug-ins are written in the Java programming language.

See *Chapter 3, FCML Overview* for more information about FCML.

Plug-In Client Applications

NETGEAR and partner developers write plug-in client applications to extend the functionality of a control point, the portal user interface, or the portal server. The client plug-ins are made up of at least one of the following libraries: OSGi JARs (services), Gadget UI (widgets), and the Canvas UI (full screen). A client plug-in can be a service, a UI component, or a service and a UI component:

- A service is an OSGi application that is deployed to the control point to add a Smart Network service to the network. See *Chapter 8, Control Point Plug-In Example* for an example control point service.
- A UI component is a web application that runs on the portal server. Customers connect to a UI component from their smart home device to communicate with Smart Network and external services. See *Chapter 9, Portal UI Plug-In Example* for an example of a UI component.

NETGEAR Control Point

NETGEAR Control Point software is installed on the control point and provides a Java-based platform for running software services. It is also a development environment for enhancing, creating, and deploying plug-in services. Services are bundled in JAR files that run on the NETGEAR Control Point.

Services consist of required services and client plug-ins. Services do things such as establish portal connections, dynamically install plug-ins, handle persistent data, control smart devices, send FCML status updates, and query FCML messages.

Required Services

- **Router**. Routes FCML messages.
- **PGW** (portal gateway). Establishes connections to the portal server.
- **OMA** (OSGi Management Agent). Provides OSGi (Open Service Gateway initiative) management. OSGi is a standards-based way to manage, configure, and inventory smart home and mobile devices. See the *OSGI Alliance home page* for more information.
- **VNS** (virtual node store). Enables a computer to interact remotely with another computer or mobile device on the Internet.

Client Plug-Ins

- **UPnP** (Universal Plug and Play). Permits networked devices in residential networks to find each other on the network to share data and communicate.
- **Z-Wave**. Allows Z-Wave-based smart devices to communicate with the ecosystem.
- **ZigBee**. Allows ZigBee-based smart devices to communicate with the ecosystem.

- **NDC** (network device client). Resides on the control point and monitors networking parameters such as IP address. Use NDC to troubleshoot or configure routers.

Control Point Architecture

The Control Point is the NETGEAR framework installed on Smart Network devices to allow deployment of applications as well as enable communications with the Portal.

The control point has the layered architecture that is shown in *Figure 3, Service and plug-in bundles running on a NETGEAR Control Point*. Security spans all layers: operating system, Java VM, OSGI, and the NETGEAR Control Point. The layers interoperate as follows:

- FCML message handle all communications between software services.
- All software services are registered as OSGi clients so they can be installed, started, stopped, and uninstalled dynamically.
- Local services should be written in Java for portability. Native code can be used, but the developer will need to create separate applications for different hardware platforms.
- Applications can use the NETGEAR Control Point SDK libraries to communicate with local APIs or to remote services (including user interfaces) in the cloud.

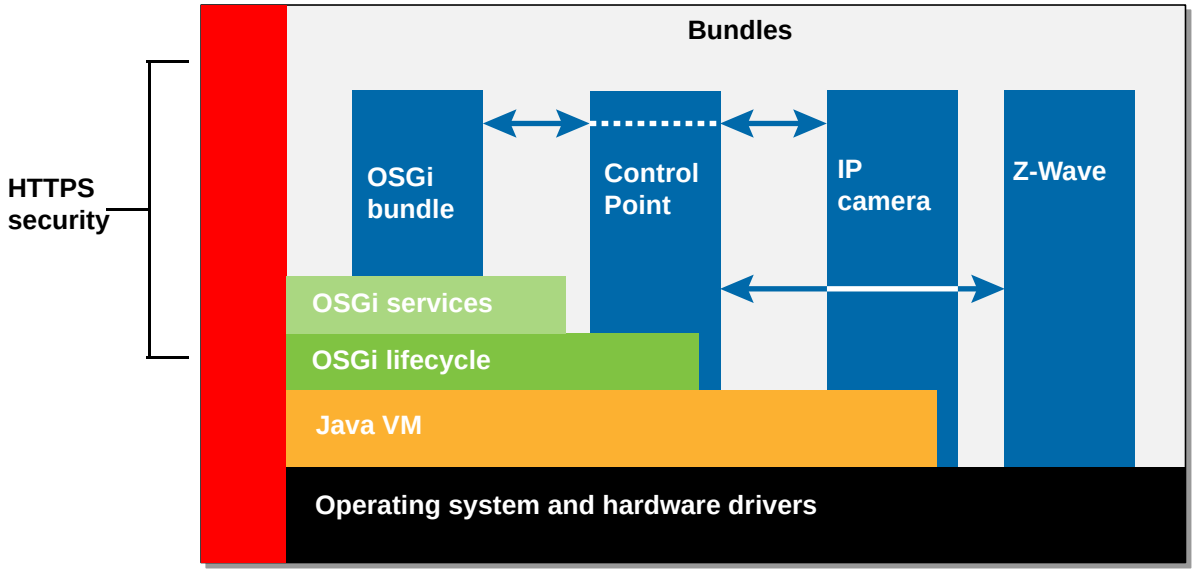

Figure 3. Service and plug-in bundles running on a NETGEAR Control Point

OSGI Service Platform. The control point uses the Apache Felix OSGi R4 Service Platform implementation.

Java VM (JVM). Devices with storage limitations use a virtual machine based on OpenJDK to run the control point. The JVM is ported from IcedTea6 and includes a JIT compiler that is ported from OpenJDK Shark.

```
java version "1.6.0_22"
OpenJDK Runtime Environment (IcedTea6 1.10.4) (Ubuntu build 1.6.0_22-b22)
OpenJDK Shark VM (build 19.0-b09, mixed mode)
```

OpenJDK SE. For the ported JVM, the OpenJDK SE classes are stripped down to fit in storage-limited devices by removing UI classes that are not commonly used in small devices. You list of supported classes is posted on the developer web site at *https://developer.netgear.com/*. You can always include classes that were stripped out in your application.

Web UI. A web page that is served by the portal server. A customer logs in to the portal server through the web-based interface to monitor their Smart Network configuration.

Client protocol (servlets). The client protocol directs FCML requests into the NETGEAR Control Point, where the services plug-in handles them. See *Client Protocol* on page 33 for more information.

.

2 How to Set Up, Package, & Deploy

Development environment setup and plug-in packaging and deployment

This chapter covers the following topics:

Development Environment Setup

You need the following software to develop a client plug-in:

- An Ubuntu 10 LTS Linux computer.
- Java platform, Standard Edition (Java SE) 6 or later.
- Eclipse IDE for Java Developers, latest version.

Ubuntu System Tools

Execute these commands to get the Ubuntu system tools.

```
sudo apt-get -y install curl git-core libesd0-dev libwxgtk2.6-dev zlib1g-dev
   build-essential libstdc++5 tofrodos
sudo apt-get -y install x-dev
sudo apt-get -y install libx11-dev
sudo apt-get -y install git-core uboot-mkimage g++-4.4 zlib1g-dev
   libncurses5-dev libreadline5-dev gperf
sudo apt-get -y install ant gawk bison flex
sudo apt-get -y install subversion
```

Java SE Installation

Execute these commands to set up your development environment.

- JDK 1.6 (sudo apt-get install sun-java6-jdk)
- export JAVA_HOME=/usr/lib/jvm/java-6-sun

You can manually set the `JAVA_HOME` environment to point to your Java SE installation:

```
readlink -f ‘which java’
/usr/lib/jvm/java-6-sun-1.6.0.15/jre/bin/java
export JAVA_HOME=/usr/lib/jvm/java-6-sun
```

Develop Your Client Plug-In

Now you are ready to develop a client plug-in. If you want examples to get you started, see *Chapter 8, Control Point Plug-In Example* and *Chapter 9, Portal UI Plug-In Example*.

Package Your Plug-In Files

Bundle the following client plug-in files into a compressed zip file:

Control Point Plug-In Only

JARs. The built OSGi-based JAR files that implement your unique services.

Portal UI Plug-In Only

- **ui/gadget.zip**. The compressed Smart Network Gadget application.
- **ui/app.zip**. The compressed Smart Network Main UI application.

All Plug-Ins

The following headings describe the info, provision, and policy files that are listed here.

- **logo.png**. The application thumbnail that the Smart Network Apps Store uses.
- **info.xml**. The metadata for your Smart Network Application.
- **provision.xml**. Optional. The metadata to provision you Smart Network Service. See the following headings for a description of this file.
- **policy.properties**. Optional. The metadata that declares your Smart Network Application access policies. See the following headings for a description of this file.

info.xml

A description of the nodes follows.

```
<?xml version="1.0" encoding="UTF-8" ?>
<app>
   <appid>NETGEAR Assigned Application ID</appid>
   <name>name</name>
   <version>1.2.3</version>
   <release-note>Release notes</release-note>
   <gadgetUrl>http://external-gadget.host.com/EntryPoint.htm</gadgetUrl>
   <appUrl>http://external-app.host.com/EntryPoint.htm</appUrl>
   <developer>Developer</developer>
   <description>Description</description>
   <thumbnail>logo.png</thumbnail>
</app>
```

appid. Smart Network Application ID received from NETGEAR.

name. The name of the Smart Network Application.

version. Smart Network Application version number.

release-note. A summary of the released version.

gadgetUrl. The value can be one of the following:

- # A URI to an external Smart Network Gadget, or
- # The entry point to the ui/gadget application, or
- # Can be ignored when ui/gadget entry point is index.htm/index.html

appUrl. The value can be one of the following:

- # A URI to an external Smart Network canvas, or
- # The entry point to the ui/gadget application, or
- # Can be ignored when ui/gadget entry point is index.htm/index.html

developer. Who developed the plug-in.

description. A short phrase that describes the application.

thumbnail. The application thumbnail that the Smart Network Apps Store uses. The logo must use *.png format and be less than 100 KB. The preferred logo size is 128 x 128 pixels.

provision.xml

Use provision.xml to define explicitly the Service Categories. The <category> name attribute identifies the Service Category identity. Each <category> has at least one <operation> element. The name attribute declares the C.O.M (FCML Client.Object.Method) operation. Mode is the operation access.

External client applications, including your Smart Network UI applications, have to declare the Service Categories that they access. Access is covered in the policy.properties section.

```
<categories>
   <category name="A">
      <operation name="client.object_type.get" mode="read"/>
      <operation name="client.object_type.set" mode="write"/>
   </category>
   <category name="B">
      <operation name="mytest.a.get" mode="read"/>
      <operation name="justAother.b.get" mode="write"/>
   </category>
</categories>
```

policy.properties

This file grants each declared Smart Network Category access to tiered Smart Network Services.

In the following example, all of the Smart Network Application components declare access to the indicated services with the specified permissions (read, write, or read-write).

```
Administration. read
Account. write
Notifications. write
Email. read
Phone. read
Address. write
Software. read
```

Deploy Your Client Plug-In

Use the developer interface to upload your bundle to the Smart Network Portal for deployment. A Smart Network administrator handles the deployment.

3 FCML Overview

NETGEAR Control Point Communication Markup Language

FCML is an object-oriented and extensible language that handles all asynchronous request and response communications between components running in Smart Network. To make it easy to learn and use, FCML is based on XML and follows the standard XML syntax and naming conventions.

This chapter covers the following topics:

Communications Flow

Client programs that run on smart home devices send FCML messages to each other and to their control point. These messages establish, change, and coordinate smart device settings and operations. *Chapter 4, Basic FCML*, explains the basic FCML sets for constructing all kinds of FCML messages. The examples show FCML messages for client plug-in and control point communications, and include how to search for object information.

control points manage all network routing communications. See *Chapter 5, Router Management APIs*, for information about how to construct network routing FCML messages.

control points and the portal server send FCML messages to each other. In addition, the portal server communicates with external modules and portal user interfaces. See *Chapter 7, Portal Client UI APIs*, for information about how to construct portal server FCML messages.

Message Structure

An FCML message is a well-formed XML document that contains one or more nodes that are wrapped in an `fcml` tag. The following message is from user interface 3 (ui3), which is running on control point 23 (cp23) to zwave1, which is also running on cp23.

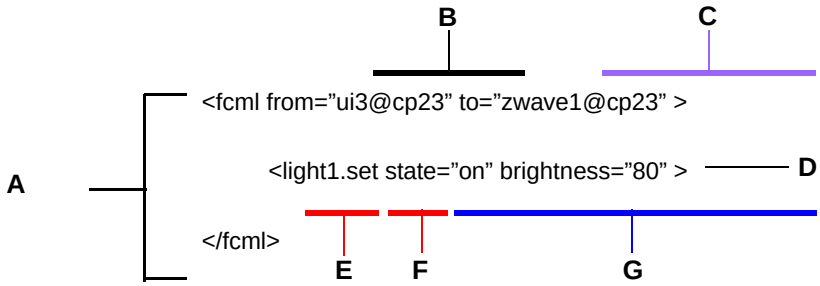

Figure 4. Parts of an FCML message

A. FCML envelope. The start (`<fcml`) and end (`</fcml>`) tags that enclose a node. The start tag has `from` and `to` parameters with attributes that indicate the address of the source and the destination program. A node contains one or more FCML methods. See *`fcml` tag* on page 23 for more information. See *Attributes and Parameters* on page 42 for more information.

B. Qualified source of message. The `from` parameter and its attribute uniquely identify the client program (ui3) and the domain address (cp23) where the message originated (ui3@cp23).

C. Qualified destination of message. The `to` parameter and its attribute uniquely identify the client program (zwave1) and the domain address (cp23) to receive the message (zwave1@cp23).

D. Node. An FCML method with parameters and attributes that define the message to send. The message defines an action to take and includes any data that is required. An FCML message can have one or more nodes that go to the same destination. See *FCML Nodes* on page 25 for more information.

E. Qualified address. Uniquely identifies the destination object as `light1`.

F. Method name. Specifies the method (or action) to take on `light1`, which is `set`.

> ***Note:*** *When a node is informational only, the action should be `is` or `says`. See `object` Type on page 46 for information about the `is` and `says` methods.*

G. Method parameter and attributes. Defines what actions to perform on the destination object (turn `light1` on to `80` watts of `brightness`).

`fcml` tag

<fcml> … </fcml>

The `fcml` tag encloses FCML messages.

Attributes

- **`from`**. Specifies the address of the client program that authored the message
- **`reply_to`**. Optionally specifies the address where to send a response message.
- **`to`**. A comma-separated list of one or more addresses of client programs to receive the message; empty means that the message is a local domain broadcast
- **`version`**. The version number of this message.

FCML message:

This message is from client 1 at device 1 (c1@d1) to client 2 at device 2 (c2@d2).

```
<fcml from="c1@d1" to="c2@d2">
   <light1.set state="on" brightness=”80”/>
   <light2.set state=”off”/>
   <light3.set state="on" brightness=”100”/>
</fcml>
```

Addresses

FCML messages are always composed with a from address and are delivered to NETGEAR Control Point Clients. Use the form `to="client@domain"` to get a message to the right client. Use the form `to="domain"` to broadcast a message to a domain.

Response Messages

Response messages are sent to the `reply_to` address when one is provided. Otherwise, response messages are sent to the `from` address that is supplied in the originating message.

Tracers

A tracer is an attribute that is used to trace requests and responses throughout the system. For example, you can use a tracer to send a request for data from one client to a number of different clients. What you do is put the same tracer on all of the messages so that you can coordinate the responses that come in at different times.

A tracer attribute name is prefaced by an underscore (_) symbol, and contains only letters, numbers, and underscores. If an FCML message contains a tracer, the tracer is required to be returned in the response to that message.

In this example, client 1 requests information about the lights on client 2 and client 3. Both requests use the same tracer so that the responses from both clients can be coordinated. Once all of the responses are received (by counting the responses with the same tracer), client 1 proceeds with its next step. See the FCML samples in *Chapter 6, Control Point APIs*, for other example uses of tracers.

Requests

```
<fcml from="c1@d1" to="c2@d1" _tracer="12345">
   <light1.get/>
</fcml>

<fcml from="c1@d1" to="c3@d1" _tracer="12345">
   <light2.get/>
</fcml>
```

Responses

```
<fcml from="c2@d1" to="c1@d1" _tracer="12345">
   <d1.c2.light1.is type="light" brightness="80" state=”on”/>
</fcml>

<fcml from="c3@d1" to="c10@d1" _tracer="12345">
   <d1.c2.light2.is type="light" brightness="0" state=”off”/>
</fcml>
```

FCML Nodes

FCML messages contain one or more nodes that are enclosed in angle brackets. For example, `<light1.set state=”on” brightness=”80” />` is a node. Node names consist of a a domain, client, object, and method specification. Domain, client, and object names start with a letter and contain only letters, numbers, and underscores.

- A method name is required, but domain, client, or object names can be omitted to indicate any domain, client, or object.

  This message deletes attribute b from device 1, client 2, object 1:

  ```
  <d1.c2.o1.is_deleted attributes="b"/>
  ```

  This message deletes attribute b from all objects from all clients running on all devices.

  ```
  <is_deleted attributes="b"/>
  ```

- You can specify more than one domain or client when there are multiple destinations.

  This message deletes attribute b from all objects running on devices d1, d2, and d3.

  ```
  <d1..d3.c2.o1.is_deleted attributes="b"/>
  ```

FCML Methods

An FCML method is what a client program calls to work on or query data. The data can be its own or the data of another client program.

Errors

Write your client programs to return an error when an FCML method fails. The error is returned with the `error` method. Some methods cannot fail and never return an error (for example, the `is` methods, and the `error` method itself). Client programs can return error codes from the following system error code list, or specify their own error codes outside the 000–999 range.

Table 1. System error codes

Code	Description	Code	Description
001	Unspecified error	123	Cannot modify element
100	Domain unreachable	130	Invalid credentials
101	No such client	131	Permission denied
110	No such object	140	Type is required for new object
111	No such method	141	Unsupported type
112	No such attribute	150	Invalid expression
113	No such element	160	Missing required attribute

Table 1. System error codes (continued)

Code	Description	Code	Description
120	Cannot add attribute	161	Missing required element
121	Cannot add element	170	Resource unavailable
122	Cannot modify attribute	180	Service temporarily unavailable

FLUID Renderer Service

The FLUID Renderer Service renders FLUID Script, either contained in an external node or passed to it in the contents of the render method tag. FLUID Script has nodes formatted similar to FCML Messages. To render a node. The children of the node are processed sequentially.

FCMLScript can run any method on any object or client.

`render` Method

The `render` method causes the renderer service to render the script source in blocks of type `text/NETGEAR Control Pointscript`. The `render` method supports the optional `oid` attribute, which is the ID of the object containing the script to render. Alternately, the `render` method explicitly specifies the block of script.

If no script tag is present in the render tag/object contents, the render method renders the entire contents of the render tag/object. This exception is for backward compatibility purposes only and might be removed from future versions of the Renderer Service. New applications should contain FLUID Script within script tags.

`script` Tag

The script tag is used to identify a block of FLUID Script source within a larger document. It supports a single attribute, type, which defaults to the only valid value: text/fluidscript.

The `script` tag identifies a block of FCMLScript source within a larger document. The `script` tag supports the `type` attribute, which defaults to the only value of: `text/NETGEAR Control Pointscript`.

Sample Script

A sample `render` method that uses the `oid` attribute to indicate that object `vns.033` contains the block of script to be executed.

```
<render oid=”vns.o33”/>
```

A sample `render` method that explicitly contains a block of script to be executed.

```
<render>
```

```
   <script type=”text/NETGEAR Control Pointscript”/>
      <lon1.o3.set state=”off”/>
      <delay duration=”1000”>
      <lon1.o4.set state=”on”/>
   </script>
</render>
```

Examples

If the following was sent to the `frs` object:

```
<fcml from="trig@core2" to="frs@core2">
<render>
<script type="text/fluidscript"/>
<core2.lon1.o16.set brightness="0"/>
<send to="lon1@core2">
<o17.set brightness="100"/>
</send>
<sleep duration="5000"/>
<if condition="core2.lon1.o2.state=='off'">
<core2.lon1.o2.set state="on" brightness="10"/>
</if>
</script>
</render>
</fcml>
```

When the `frs` object begins to process the render tag, it first sends this message:

```
<fcml from="trig@core2" to="lon1@core2">
<core2.lon1.o16.set brightness="0"/>
</fcml>
Next, it would send:
<fcml from="trig@core2" to="lon1@core2">
<core2.lon1.o17.set brightness="100"/>
</fcml>
```

Next, the `frs` object pauses for five seconds and then sends this message:

```
<fcml from="trig@core2" to="lon1@core2">
<get select="state"/>
</fcml>
```

If the Echelon driver responded with this:

```
<fcml from="lon1@core2" to="trig@core2">
<core2.lon1.o2.is state="off">
</fcml>
```

Then the `frs` object sends this message:

```
<fcml from="trig@core2" to="lon1@core2">
<core2.lon1.o2.set state="on" brightness="10"/>
</fcml>
```

FCMLScript

Statements

FCMLScript statements are always executed in the order they appear in the document.

if_else Statement

`if_else` statements contain only `if` and `else` statements. Control breaks out of an `if_else` statement after the statement executes the contents of its `if` or `else` statement.

if Statement

The contents of `if` statements are executed when the expression in the `condition` attribute evaluates to `true`.

else Statement

The contents of `else` nodes are executed when the expression in the `condition` attribute of the `if` node evaluates to `false`.

render Statement

Start another `render` thread using the contents of the `render` node, or the contents of the object that is specified by the `oid` attribute if present. Watch out for infinite recursion.

send Statement

`send` generates an FCML message that is from the renderer service. The `send` statement and the `fcml` tag are synonymous in syntax, usage, and behavior. The `send` tag exists for backwards compatibility with a previous version of FCMLScript.

sleep Statement

The `sleep` statement causes a pause in execution before the next child is rendered. The `duration` attribute specifies the length of the pause in milliseconds. The `delay` statement is a deprecated synonym for `sleep`, intended for backward compatibility with a previous version of FCMLScript.

Attribute Value Substitution

The value of an attribute can be substituted in a string using the following form, where ATTRIBUTE is an attribute ID:

```
"${ATTRIBUTE}"
```

Attribute substitution works only within quotes. For example, after the following script executes, the name attribute of the vns.o2 object is set to another one.

```
<vns.o1.set name="one"/>
<vns.o2.set name="another ${vns.o1.name}"/>
```

local Object

The renderer service supports an object named local that is subjective to each interpreter instance (thread) so that scripts can create and use local variables. For example, after the following script executes, the count attribute of the vns.o1 object is set to 5.

```
<local.set i="5"/>
<vns.o1.set count="${local.i}"/>
```

Logs and Alerts

A client plug-in can send an FCML message that is entered into a log. FCML log entries are normal FCML objects of type log. All log objects are virtual. All log objects exist in the virtual node store (VNS) service, are archived on the portal, and support the following attributes:

- **type**. The type of the alert. Values are log or alert.
- **urgency**. User alert behavior. Values can be none, low, medium, and high.
- **level**. Categorize non-urgent log entries with a level from 1 to 5.
- **text**. A string that provides the details of the log entry.
- **read**. A Boolean value that is true when the log entry has been read.

Log Entry Creation

```
<fcml from="ui1@cp3" to="vns@cp3">
   <new type="log" urgency="high" text="Motion detected on sensor 4"/>
</fcml>
<fcml from="vns@cp3">
   <vns.o25.is_new type="log" urgency="high"
                   text="Motion detected on sensor 4"/>
</fcml>
```

Alerts

An `alert` is a log entry with an `urgency` level that is *not* `none` or `null`. UI clients and the portal listen for VNS broadcasts that a new log object is created. The behavior of the user interfaces and the portal server depend on the `urgency` setting, as follows:

1. High urgency:
   - UI pops up an urgent message.
   - UI displays `an urgent message waiting` indicator.
   - Portal sends an email or SMS as configured for high urgency.
   - Portal saves a copy of the log entry.
2. Normal urgency:
   - UI displays a message waiting indicator.
   - Portal sends an email or SMS configured for normal urgency.
   - Portal saves a copy of the log entry.
3. Low urgency:
   - UI displays a message waiting indicator.
   - Portal sends an email or SMS configured for low urgency.
   - Portal saves a copy of the log entry.
4. No (none) urgency. The portal saves a copy of the log entry.

NETGEAR Control Point Logging

NETGEAR Control Point log messages are streamed to the ROOT/messages file. The default log level is INFO, which means ERROR, WARN, and INFO messages are streamed to the log file. You can change the log level to DEBUG so that you get only debugging-related messages in the log file.

Look for whether NETGEAR Control Point successfully connected to the simulated portal. You also see messages about clients that are connecting to core and other messages. After the first minute, the rate of messages slows down as the clients establish connections.

```
11:57:21.240 INFO : Authenticating with portal
11:57:21.328 INFO : Connecting to portal http://myportal.com/fcp/init
11:57:21.398 INFO : Connected to portal
```

Log Errors

If there is a problem, you see ERROR messages in the log. Common error messages include the following:

- **Failure to connect to or activate with portal**. To resolve this failure, check that the portal address in the fluid-core.conf file is correct, and that the portal is running at that address.
- **Failure to set up serial communication, or rxtx (if using Z-Wave)**. To resolve this failure, check that you specified the correct location of the Z-Wave dongle in the config.properties file.
- **Failure to start certain bundles**. To resolve this failure, check any new JAR bundles that you test. Check the JAR bundles for a failure to start or a failure to connect to NETGEAR Control Point.

Control Point Router Types

There are two types of control points: domain and gateway. A domain is the group of smart home devices that are all connected to the same control point. You can configure a home or business to have two or more control points to manage separate domains. The same network can have two or more gateway control points to forward messages between the domains.

Domain Routers

The domain control point is responsible for the correct delivery of FCML messages within its domain and to the gateway control points for the network. Client programs pass messages to the domain control point, which then delivers the messages to the destination addresses. The destination addresses are specified in the `to` attribute of the `fcml` tag. A message that has no `to` attribute is called a *broadcast* message because it is delivered to all client programs within the domain.

The domain control point delivers messages as follows:

- Broadcast messages and messages with local domain broadcast destinations are delivered to all client programs within the local domain.
- Messages with local domain destinations are delivered to the named client program within the local domain, if present.
- Messages with remote domain destinations (including remote domain broadcasts) are delivered to all gateway client programs within the local domain.

The domain control point does not do the following:

- Does not deliver a message to the client program that sent the message.
- Does not modify messages in any way.
- Does not route a message with a `from` address that differs from the address of the originating client program (except for gateway clients).

Communicate with the Domain Control Point

All domains have a pseudo-client named `router` that enables FCML communication directly with the domain control point. Services can send messages to the `router` client to query the domain control point for information about the domain itself. The domain information includes a list of clients and diagnostic information.

For example, the following FCML message queries the domain control point for a list of all client programs executing in the core2 domain.

Request to router@core2

```
<fcml from="ui1@core2" to="router@core2">
   <get/>
</fcml>
```

Response from router@core2

```
<fcml from="router@core2" to="ui1@core2">
   <core2.router.router.is type="router" domain="core2"/>
   <core2.router.lon1.is type="driver"/>
   <core2.router.vns.is type="store"/>
   <core2.router.ui1.is type="ui"/>
   <core2.router.ui2.is type="ui"/>
</fcml>
```

Gateway Control Points

Gateway control points forward messages between one domain and another. However, gateway control points do not forward messages that are directed solely to the gateway itself. Gateway control points also do not forward messages with recipients in the originating domain or modify forwarded messages in any way.

When gateway control points process messages in their receive queue, they do not have to be connected to a remote domain. Gateway control points drop messages that are destined for domains they cannot immediately reach as necessary when their queues are full.

Control Point Configuration Page

NETGEAR Control Point serves the /config page that provides various utilities. To display this page, type the IP address of the router on port 5050 into your browser address box. For example, **192.168.1.1:5050/config** The /config page is tabbed with the title *Smart Network Control Point Configuration*.

Tabs description

- **OSGi tab**. Provides a list of bundles, their versions, and their current state, in addition to a button to stop/start each.

- **Logging tab**. Provides access to the log file, in addition to the log file from the immediately previous run.
- **Cameras tab**. Provides list of cameras that are known to NETGEAR Control Point.
- **Z-Wave tab** and **ZigBee pages**. Provide a list of known devices.

Client Protocol

The client protocol is a five-channel system that runs on the portal server and the control points. The channels handle client authentication and initialization in preparation for sending and receiving FCML messages. The authenticate channel runs only on the portal server. The other channels run on the portal server and on the control points.

- `activate` channel
- `authenticate` channel
- `init` channel
- `send` channel
- `receive` channel

> ***Note:*** *Smart Network has a wrapper for Java plug-ins to handle everything that these channels do so that your code does not have to make calls to the channels. If you use a technology other than a Java plug-in, keep reading because you need to know how to make calls to the channels.*

The channels are servlets that are invoked from a client program by channel URLs. The channel URLs are relative to the URL that the UI software is served from, within a top-level directory named `/fcp`. For example, if the UI is served from www.smartnetwork.net/argon/, then the FLUID client protocol (FCP) authenticate channel would be at www.smartnetwork.net/fcp/authenticate.

The `activate` channel is used only by the portal gateway client running on the control point. The control point gets its domain activation credentials (domain name, domain password) and expiration date through a conversation on the `activate` channel.

The `init`, `send`, and `receive` channels are protected channels. Only clients with valid sessions can use these channels.

The `authenticate` channel runs only on the portal server. It is open and used to obtain client sessions for access to the protected channels and any other protected resources.

Channel Connection Flow

The communication flow across the five channels is depicted in the following figure. The text that follows the figure describes the flow in words.

Note: *the init comes before the authentication to enable the application to work seamlessly whether it is connecting to the Portal server or to the Control Point. If you follow this best practice your UI will work locally as well as remotely.*

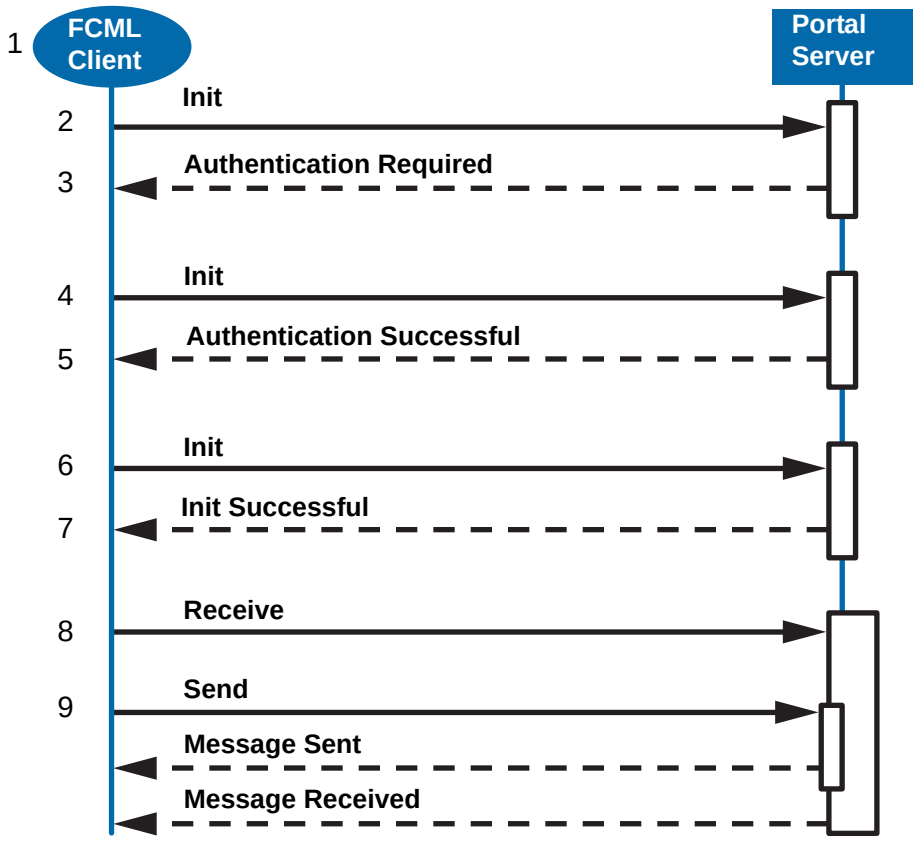

Figure 5. Communication flow between the portal server and FCML clients

1. Start with no channels connected.
2. Send an `init` message without authentication credentials on the `init` channel.
3. Receive negative authentication confirmation on the `init` channel.
4. Send an `init` message with authentication credentials on the `init` channel.
   a. Connect to the `authenticate` channel.
   b. Receive an authentication challenge.
   c. Respond to the challenge with credentials.
   d. If authentication fails, go to step 4.c.
5. Receive a positive authentication confirmation on the `authenticate` channel.
6. Resend the `init` on the `init` channel.
7. Receive a successful `init` response on the `init` channel.
8. Connect to the `receive` channel.

9. Send an FCML message on the `send` channel, which connects to the `send` channel.
   a. On the `send` channel, a `receive` confirmation the message was sent.
   b. On the `receive` channel, the response message is received.

Client Sessions

To use the protected channels (`init`, `send`, `receive`) a client has to have a valid session. Authentication establishes a valid session (tracked by a session ID cookie). As long as the client's session is valid, the client can communicate on the protected channels.

Sessions can become invalid due to inactivity. If no requests are made on any protected channel for 10 minutes during some session, the server invalidates that session due to inactivity.

Sessions can also be invalidated administratively or invalidated when the client uses the `logout` command on the authenticate channel.

Authentication

If a client makes a request on a protected channel without a valid session, the server responds with an `unauthorized` message. NETGEAR Control Point clients might need to authenticate with a NETGEAR Control Point server before they can send or receive FCML.

There are three supported authentication schemes:

- **`basic`**. A clear-text user name and password for logging in.
- **`token`**. An opaque string (hash) that is used by user interfaces under a single sign-on schema when a third party has already authenticated the user.
- **`header`**. A user ID is presented in an HTTP header. This scheme is used only when the control point identity has been verified by a third party. For example, when a load balancer that uses digital certificates is sent to a portal that is protected by the authenticating party.

Basic Authentication

From within a client program, send an XML message to the portal server to authenticate with the appropriate user name and password:

```
<authenticate type="basic" username="me" password="mypass" />
```

The portal server returns a confirmation to the client program that authorizes it to run in the basic protected channel.

```
<authenticate type="basic" authenticated="true" />
```

Token Authentication

From within a client program, send an SML message to the portal server to authenticate with a token.

```
<authenticate type="token" token="12a8d7f88dd998e9a899d" />
```

The portal server returns confirmation to the client program that authorizes it to run in the token protected channel.

```
<authenticate type="token" authenticated="true" />
```

Header Authentication

From within a client program, send an XML message to the portal server to authenticate with a header.

```
<authenticate type="header" />
```

The portal server returns confirmation to the client program that authorizes it to run in the header-protected channel.

```
<authenticate type="header" authenticated="true" />
```

De-Authentication

The portal server deauthenticates a client program by sending a logout message.

```
<logout/>
```

In this example, the UI reconnects to its receive channel and receives a message that it is no longer authenticated.

```
<authenticate result="401" description="Unauthorized" />
```

`init` Channel and Client Name

Clients obtain their client name through a conversation on the `init` channel using `init` messages.

Request Client Name

```
<init [type="TYPE"] [name="STATICNAME"] [fcmb="true"] />
```

`TYPE` is the client type. If no type is provided, the router assigns the type `ui` to the client. `STATICNAME` is the client name that is required by the client. This is important for gateways, drivers, and stores. Clients of type `ui` *should not* request a static name. The domain router *has to* return an error response if `STATICNAME` cannot be assigned to the client.

The `fcmb` attribute lets the server know that this client can receive batches of FCML messages using the NETGEAR Control Point Communications Batch Protocol. Whether or not the server sends batches to batch-capable clients is up to the server.

Receive Client Name

```
<init name="CLIENTNAME" domain="CLIENTDOMAIN" type="CLIENTTYPE"
   read_timeout="READTIMEOUT" [fcmb="true"] />
```

- `CLIENTNAME` is the name that is issued to the client.
- `CLIENTDOMAIN` is the domain in which the client name was issued.
- `CLIENTTYPE` is the client type the client name was issued to.
- `READTIMEOUT` is the number of seconds to wait to receive a message before assuming that a communications failure has occurred. The default is 122 seconds (2 minutes plus 2 seconds). Some clients do not have the ability adjust their `read_timeout`. For example, the Flash VM always times out after 30 seconds.
- The `fcmb` attribute when present and set to `true` indicates that the server can receive batches of FCML messages using the NETGEAR Control Point Communications Batch Protocol.

Error Initializing

This FCML message means that an error occurred during initialization:

```
<init error="true" code="14" description="Something went wrong." />
```

State Refresh

Sometimes there are situations where the outgoing message queue for a client exceeds the capacity of the queue. This situation could happen if the client is unable to retrieve messages as fast as they are queued. When the queue capacity is exceeded, the server discards the messages in the queue. The server then sends a `refresh` message to the client. The message lets the client know that messages have been lost and to refresh its state (cache). No more messages are queued for that client. To reestablish communication the client has to obtain a new client name through the `init` process.

No Multiple Clients Per Session

FCP does not support multiple clients per session. For example, when a customer opens a second UI in another tab of the browser, FCP invalidates (logs out) any existing clients in the same session.

`receive` Channel Time-Out

A request on the `receive` channel does not return until a message is available or the request times out. The server times a request out after a predetermined time interval. The default and suggested time-out interval is 2 minutes. The client can time out before the server. For example, the Flash VM always times out after 30 seconds.

A timeout on the receive channel returns an empty response and is not an error condition. A timeout on the receive channel does not `invalidate` a session.

Activity Detection

Normally a client reconnects immediately to the `receive` channel after a completed message delivery or `receive` channel time-out. The reconnection refreshes the active status of the client and keeps the session from becoming invalid due to inactivity. If a client

does not reconnect within 10 seconds, the session expires and the `active` attribute of the client is set to `false`.

`init` Examples

When the UI program starts, it directly connects to a control point, gets a client ID, and then queries the portal server.

```
// Send an XML init message on the init channel
<init type="ui" />

// Receive an XML init response message on the init channel
<init name="ui3" domain="cp1" type="ui" />

// Send an FCML get message on the send channel
<fcml from="ui3@cp1" to="router@cp1">
   <get />
</fcml>

// Receive an FCML response on the receive channel
<fcml from="router@cp1" to="ui3@cp1">
   <cp1.router.router.is type="router" domain="cp1"
   friendly_name="home" image="home.jpg" />
   <cp1.router.lon1.is type="driver" />
   <cp1.router.ui1.is type="ui" />
   <cp1.router.ui2.is type="ui" />
   <cp1.router.ui3.is type="ui" />
</fcml>
```

In this example, the UI starts, connects to the portal server, authenticates the client, picks an control point, and gets a client name in that domain.

```
// Send an XML init message on the init channel
<init type="ui" />

// Receive a negative authentication response on the init channel
<authenticate result="401" description="Unauthorized" />

// Send an XML authentication request on the authenticate channel
<authenticate types="basic" />

// Receive an authentication challenge on the authenticate channel
<authenticate type="basic" />

// Respond to the authentication challenge on the authenticate channel
```

```
<authenticate type="basic" username="me" password="mypass" />

// Receive a confirmation of authentication on the authenticate channel
<authenticate type="basic" authenticated="true" />

// Resend the init message on the init channel
<init type="ui" />

// Receive an init response on the init channel
<init name="ui2" domain="portal" type="ui" />

// Send an FCML get message on the send channel
<fcml from="ui2@portal" to="router@portal">
   <get />
</fcml>

// Receive an FCML response on the receive channel
<fcml from="router@portal" to="ui2@portal">
   <portal.router.cp1.is type="domain" friendly_name="vacation home"
                      image="vacation.jpg" active="false" />
   <portal.router.cp2.is type="domain" friendly_name="mom's house"
                      image="mom.jpg" active="true" />
   <portal.router.cp3.is type="domain" friendly_name="home"
                      image="home.jpg" active="true" />
</fcml>

// Send an FCML domain join intention on the send channel
<fcml from="ui2@portal" to="router@portal">
   <portal.router.cp2.join client="ui2@portal"/>
</fcml>

// Receive an FCML domain join confirmation on the receive channel
<fcml from="router@portal" to="ui2@portal">
   <portal.router.cp2.is type="domain" joined="ui2@portal"/>
</fcml>
```

In this example the portal client starts and connects to the portal server.

```
// Send an XML init message on the init channel
<init type="gateway" />

// Receive a negative authentication response on init channel
<authenticate result="401" description="Unauthorized"/>
```

```
// Send an authentication request on auth channel
<authenticate types="basic" />

// Receive an authentication challenge on the authenticate channel
<authenticate type="basic" />

// Respond to the authentication challenge on authenticate channel
<authenticate type="basic" username="cp2" password="cp2pass" />

// Receive confirmation of authentication on the authenticate channel
<authenticate type="basic" authenticated="true" />

// Resend the init message on the init channel
<init type="gateway" />

// Receive an init response on the init channel
<init name="cp2" domain="portal" type="gateway" />
```

Basic FCML

4

The basic objects and search

This chapter explains the basic FCML sets for constructing all kinds of FCML messages. The examples show FCML messages for client plug-in and control point communications, and include how to search for object information. Use the information in this chapter and the following chapters to add FCML communications to your client plug-ins.

This chapter covers the following topics:

FCML Objects

FCML objects are the basic data structure in NETGEAR Control Point communications. Client programs create FCML objects when they call the `new` or any of the `set` methods. Objects have identifiers so they can be reached, and are made up of attribute data and element data. Attribute and element data can be queried and manipulated by FCML methods.

Identity

An FCML object needs an *identifier* (name) so it can be reached by client programs. When a client program creates an FCML object by calling the `new` method, the object is assigned a name and is owned by the client program that created it. Sometimes an object is created with a client program call to one of the `set` methods. In this case the object name is specified in the call to `set`, and the object is owned by the client program that called the `set` method. Either type of object can be deleted by a call to the `delete` method. The client program executes the call to `delete`, but it can be at the request of another client program.

> ***Note:*** *Do not reuse object identifiers after the object has been deleted. Doing so results in an error message because the object no longer exists.*

Attributes and Parameters

Objects have named attributes with parameter values. An attribute is a quality or characteristic that is inherent to the object. For example, the attributes of the `fcml` tag are `from` and `to`, which specify the sending and receiving client programs. Attribute names start with a letter and contain only letters, numbers, and underscores (_). By convention, attributes are named completely in lowercase with words separated by an underscore.

Elements

Objects can contain elements, which are XML sub-nodes that are used to represent binary (CDATA) or recurring data within an object. Recurring data is represented as multiple XML nodes with attributes and potentially nested elements. Binary data is included as the value of the object itself. The following example shows `<ref>` elements that contain the IDs of other objects in the system. It also contains a value following the refs. This section represents any intrinsic value the object might have beyond it's attributes. In most cases this is data that cannot or should not be parsed into XML, such as binary or free-form text.

```
<fcml from="c2@d1" to="c1@d1">
   <_result_d1_c1_1.is_new type="result" path="light" limit="3"
                           start="1" count="6">
   <ref>o1</ref>
   <ref>o2</ref>
   <ref>o3</ref>
```

```
   <![CDATA[THIS SECTION CONTAINS ANY BINARY OR UNPARSED TEXT DATA]]>
   </_result_d1_c1_1.is_new>
</fcml>
```

Elements need an ID attribute, referred to as the element ID, to uniquely identify the element within the object. Element identifiers are assigned either by the client that adds the element, or automatically by the owning client for new elements specified without an ID.

Element IDs start with a letter and contain only letters, numbers, and underscores (_). The ordering of top-level elements within an object is client-specific and not defined by a specification. Clients should make no general assumptions about the ordering of top-level elements and have to provide their own mechanism for ordering elements if needed. For example, you might use an index attribute to order elements at the application layer.

Elements can have additional attributes and can contain other elements. Contained subelements do not need to have ID attributes.

How Client Programs Describe Objects

Client programs describe their objects to other client programs with `is` methods (`is`, `is_all`, `is_deleted`, `is_new`, and `is_pending`). When to describe objects, which `is` method to use, and whether to unicast or broadcast object descriptions depends on the client and on the situation.

> ***Note:*** *A unicast is when messages are sent to a single network destination that is identified by a unique address. A broadcast is when the same messages are sent to all possible destinations.*

Follow these guidelines to help you decide when and how to describe objects in your own class implementations.

1. A client has to unicast a description of an object in response to a `get` method request.
2. A client has to broadcast or unicast a description of all object changes to the object that created it if the object is referenced by a result object.
3. A client should broadcast a description of all object changes if it supports a moderate number of objects, especially small objects, or objects that do not change often. For example, when a switch object changes power states. The broadcast enables other clients to stay live without polling.
4. A client should describe deleted attributes and elements that use the `is_deleted` method. Alternately, it can use the `is_all` method if the object has changed enough that it would be more efficient to redescribe the object. For example, you might use the `is_all` method when a complex object is completely rewritten by the `set_all` method.

Send an FCML Message

The client program makes an HTTP request to establish communication with the portal server. The HTTP request has the FCML message in its body and includes a URL in the format http://www.smartnetwork.com/fcp/send. The portal server recognizes the requestor by the `from` field of the FCML message. The receiving servlet responds that it received the message with the HTTP status OK. The portal server routes the FCML message to the In queue, and on to the portal client to which the message is addressed.

For example, a message for the locations client is addressed to the enterprises portal client at the portal domain: `enterprises@portal`. The FCML message goes into the portal server In queue for processing. You can wrap the FCML message in a `String` object or in an `FCMLDocument` object. The response is always an `FCMLDocument` object.

The following examples show how to form the request as a `String` object and as an `FCMLDocument` object.

Request using a String object

The `String` object code starts on line 4.

```
1  URL url = new URL("https://smartnetwork.netgear.com/fcp/send");
2  URLConnection connection = (HttpURLConnection) url.openConnection();
3  PrintWriter ouputstream = new
      PrintWriter(httpConnection.getOutputStream());
4  String body = "< FCML from=\"netrouter@cp4\" to=\"enterprises@portal \" >
      <get /> </FCML>"
5  ouputstream.write(body);
6  ouputstream.flush();
7  httpConnection.connect();
```

Request using an FCMLDocument object

The `FCMLDocument` object code starts on line 4.

```
1  URL url = new URL("https://smartnetwork.netgear.com/fcp/send");
2  URLConnection connection = (HttpURLConnection ) url.openConnection();
3  PrintWriter ouputstream = new
      PrintWriter(httpConnection.getOutputStream());
4  FCMLDocument doc = new FCMLDocument();
5  FCMLAddress from = new FCMLAddress("netrouter", "cp4");
6  doc.SetFrom(from);
7  doc.addTo("enterprises", "portal");
8  doc.addNode("get");
9  String body = doc.asFCML();
10 ouputstream.write(body);
11 ouputstream.flush();
12 httpConnection.connect();
```

Receive an FCML Message

The portal server processes the message, creates an FCML response message, and puts the response in the client's queue.

To get the response, the client program sends an HTTP request that specifies a URL in the format portal/Domain/fcp/receive. When the portal server receives the request, it checks the client queue for messages addressed to that requestor. If the portal server finds one or more messages, it puts them in the response body and sends the response.

Response using an FCMLDOcument object

```
URL url = new URL("https://smartnetwork.netgear.com/fcp/receive?n=cp3");
URLConnection connection = (HttpURLConnection) url.openConnection();
httpConnection.connect();
FCMLDocument doc = new FCMLDocument(httpConnection.getInputStream());
```

Common Methods

Portal clients and objects, but not types, all implement the following methods. See *Chapter 4, Basic FCML* for details on these methods.

> ***Note:*** *Types do not implement the common methods because any number of objects can have the same type. You can call the method on a specific object of a certain type, but not on the type itself.*

- `get` Method described on page 46
- `set` Method described on page 48
- `new` Method described on page 49.
- `delete` Method described on page 46
- `is_deleted` Method described on page 47
- `is_new` Method described on page 47
- `is` Method described on page 46
- `is_all` Method described on page 47

`object` Type

All objects are type `object`. The `object` type defines and implements the basic methods for describing and manipulating objects through FCML messages. The `type` attribute for every object has to be `object` or an extension of `object`.

`delete` Method

The `delete` method deletes attributes and elements of the object, or deletes the object itself. If no attributes or elements are specified, then the object itself is deleted. Clients should describe deleted attributes, elements, and objects in the `is_deleted` method.

Attributes

`attributes`. A comma-separated list of names of attributes to delete.

`elements`. A comma-separated list of IDs of elements to delete.

Examples

```
<o1.delete attributes="a,b"/> // Deletes attributes a and b
<o1.delete elements="e1"/> // Deletes element el
<o1.delete/> // Deletes object o1
```

`get` Method

The `get` method requests a client program to provide a description of the specified attributes.

Attribute

`attributes`. A comma-separated list of names and attributes that describe the object and its elements. If the `attributes` attribute is specified in the call to the `get` method, the client program should describe only the attributes listed. This attribute should not include elements that use the `is` method. Otherwise, the client response has to describe the object completely using the `is_all` method.

`is` Method

The `is` method describes the specified object cumulatively. Cached client programs need to update any of the stored attributes and elements mentioned in the call to the `is` method before they return a response to the calling client. When a cached client program updates stored elements, newly mentioned attributes should be added, mentioned stored attributes modified, and stored content replaced if mentioned. There should be no effect on unmentioned attributes and elements.

> ***Note:*** *The `is` method cannot describe deleted elements or attributes. Clients should use the `is_deleted` method to describe deleted elements or attributes.*

`is_all` Method

The `is_all` method describes the specified object completely.

Cached client programs should replace any stored representation of the object with the description provided.

`is_deleted` Method

The `is_deleted` method describes deleted attributes, elements, or a deleted object. When no attributes or elements are returned, then the object itself has been deleted.

Attributes

`attributes`. A comma-separated list of names of attributes that have been deleted.

`elements`. A comma-separated list of IDs of elements that have been deleted.

`is_new` Method

The `is_new` method announces and completely describes a new object.

`is_pending` Method

The `is_pending` method describes a pending change to one or more attributes or elements. This method is followed by either an `is` method that mentions the pending attributes or an `is_all` method. The `is_all` method completes the state of the pending change even when the attribute and element values did not change.

`says` Method

The `says` method conveys a message that has information not represented in the object's data model of attributes and elements. Client programs should not use this method to convey state information that can be naturally represented as attributes and elements.

For example, the information "the password was just changed" should not be represented as a change to a `password` attribute because passwords should not be exposed. It could be represented with a `password_changed` attribute that pulses from `false` to `true` and then back to `false` when the password changes. However, this is awkward and saddles the object with a transitory attribute that does not convey information about the object's current state. In this case, the best way to represent the transitory information would be to use the `says` method. Client programs can include additional attributes with details about the message.

Attributes:

`key`. A short client-defined code uniquely identifying the meaning of this message.

`description`. An optional plain-language description of this message.

`set` Method

The `set` method modifies the object's attributes and elements. Client programs should use the `is` method to describe object changes that are caused by the `set` method. If the specified object does not exist, a new object is created with the specified name, attributes, and content, subject to the same requirements as the client `new` method.

If the specified object exists, the mentioned attributes already existing on the object are updated. Mentioned attributes not already existing on the object are added to the object. Mentioned elements that already exist on the object (determined by element ID) are updated. During the update newly mentioned attributes are added, existing attributes, if mentioned, are modified, and existing content, if any is mentioned, is replaced. Mentioned elements that do not specify an existing element ID are added to the object. The owning client has to assign IDs for added elements when its IDs are not specified by the creating client. There is no direct effect on unmentioned attributes or elements. The `type` attribute cannot be mentioned.

> ***Note:*** *This method cannot be used to delete attributes or elements from an object. Clients have to use the `delete` method to delete elements or attributes.*

`set_all` Method

The `set_all` method sets all the specified object's attributes and elements. Clients should describe object changes that are caused by the `set_all` method with the `is_all` method.

If the specified object does not exist, a new object can be created with the specified name, attributes, and content. The new object is subject to the same requirements as the client program's `new` method.

If the specified object exists, all existing attributes and elements are replaced by the ones mentioned. The client that created the object has to assign IDs to elements that were added. The `type` attribute should not be mentioned.

`client` Type

The `client` type represents an FCML client.

`error` Method

The `error` method reports an error that contains the original method that generated the error, an error code, and a description of the error.

Attributes

`code`. The error code.

`description`. The error description.

`details`. Optional. Any additional details about the error.

get Method

The get method requests a description of all objects that match the specified criteria in a where expression. The get method does not examine the content of objects when it evaluates the where expression, but instead examines only object attributes. If no objects match the criteria, the client should not respond with any message (unless the message is traced). The get method should not trigger any broadcast message.

If the attributes attribute is specified, the client program should describe only the attributes list, and no elements that use the is method. Otherwise, the client program has to completely describe the objects using the is_all method.

Attributes

attributes. A comma-separated list of attributes to describe.

path. An XPath-like expression to select objects to return (alternative to where attribute).

where. An expression that evaluates to true for the node to be returned.

where Expressions

Attribute names conform to the following forms.

- A. The attribute (A) of this object.
- O.A. The attribute of the specified object (O).
- C.O.A. The attribute of the specified object wned by the specified client (C).
- D.C.O.A. The attribute of the specified object owned by the specified client running in the specified domain (D).

Permitted conditional operators are and, or, >, >=, ==, and !=. Parentheses ( )can be used to group conditions. Only objects can have attributes; clients cannot have attributes. The letters A and OA refer to object A within the client that is interpreting the where clause.

new Method

The new method creates a new object that is owned by the client that calls the new method. All attributes and elements mentioned should be stored on the new object. The type attribute is required for new objects. Clients should announce new objects using the is_new method.

search Method

The search method sends a request for a search to another object. The object that receives the search request, conducts the search according to the attributes (settings and search criteria) specified in the request.

Attributes

`limit`. The maximum number of objects to reference at a time in the `result` object's content; default is `none` (no limit). For example, if you set the `limit` to 3, the `result` object returns three results at a time.

`order`. A comma-separated list of attributes describing how to order objects in the search result. Results can be ordered by top-level object attributes. The bang (!) prefix means descending alphabetic order. The default is client-dependent.

`path`. The search criteria. The syntax is similar to the syntax used in the XPath query language. The `path` is an expression that describes the objects to include in the search result. The default is `*` to return all objects.

> ***Note:*** For the `path` attribute, the `search` method sees objects with their object ID and type swapped. The object type is now the node name, and the `id` attribute now holds the object ID, which is more convenient in a search for objects of a specific type. For example,use `path "log"` instead of `[@type='log']` to search for `log` objects.

`result`. The ID of the result object to be created. The `search` method always produces a result object, unless there is an error, in which case an error response is generated. Even if no objects match the search criteria, a result object is still generated with count="0" and no ref elements.

Result objects are named with the format: _result_DOMAIN_NAME_SUFFIX.

`DOMAIN` and `NAME` are the domain and name of the client that does the search, and `SUFFIX` is a string that identifies the result object to the client. The object named cannot exist. It is an error to specify an existing object.

`start`. A 1-based index of the first object referenced in the `result` object's content. The default is 1 to reference the first object in the result set. If the value of `start` is larger than the number of objects in the result set, no error is returned. The number of objects in the result set is indicated by the count attribute value of the `result` object. However, the `result` object will contain no `ref` elements.

`result` Type

The `result` type represents objects that contain references to other objects (`ref` elements) in a set. A call to the `search` method creates a `result` object. The contents of a `result` object are managed in response to changes in the underlying object collection. When you modify the `start` or `limit` attributes of the `search` method, the content in the `result` object is modified to reflect the new `start` index and reference `limit`.

Client programs have to call the `get` method of the `result` object to retrieve the objects that are referenced within the `result` object. However, if any objects referenced in the result set change, an `is` has to be sent by those objects.

Changing the `start` attribute on a `result` object updates the contained references to reflect the new starting item number. This update is useful for paging through a large result set. If the value of `start` is larger than the number of objects in the result set (`count`), no error is returned. Because paging a `result` object causes a complete replacement of all elements, clients should use `is_all` to describe page changes (instead of using `is_deleted` / `is`).

All `result` objects consume server-side resources and should be deleted by the client when they are no longer needed. All `result` objects that belong to a client program are deleted when the client program disconnects.

Attributes

`count` (read-only). The total number of objects in the result set.

`limit`. The maximum number of objects referenced in the content.

`owner` (read-only). The client (*`NAME@DOMAIN`*) that owns the `result` object.

`order` (read-only). A comma-separated list of attributes describing the order of objects in the result set. The bang (!) prefix means descending.

`path` (read-only). An XPath-like expression that describes the objects to be searched for.

`start`. A 1-based index of the first object referenced in the content.

`ref` Element

The `ref` element contains a NETGEAR Control Point object ID. The ID does not have to be fully qualified, and typically provides the object name only.

Attribute

`type`. The type of the object. Types can be `object`, `client`, `result`, or `device`.

`device` Objects

Device drivers interact with hardware devices through `device` objects. A `device` object has a number of attributes and can implement interfaces. Devices can also be controlled through interfaces, which are collections of FCML custom methods.The various device types have interfaces that they can implement.

Data Types

Device attributes support the following data types:

`string`. One or more characters.

`integer`. A positive or negative natural number.

`float`. A positive or negative double precision real number.

`onoff`. An enumeration that accepts either on or off.

`percent`. A floating point number from 0 to 100).

`degrees`. A floating point number from –360 to 360.

`color`. An enumeration that accepts one of the following values: red, orange, yellow, green, blue, indigo, or violet.

`temperature`. A floating point number that represents degrees of celsius.

`device` Type

The `device` type is the base class for all devices. The easiest way to control a device is to set an attribute. For example, many extended device types (described in the following table) have a `state` attribute that represents the power state of the device. User interfaces can use the FCML `set` command to change the value of the state attribute to turn the device on or off.

Device implementations have to support all the following attributes:.

Table 2.

Type	String (read)	Description
implements	List (read)	List of implemented interfaces
Active	Boolean (read)	Device is readable and controllable
Make	String (read)	Device is readable and controllable
Model	String (read)	Model of the device
Name	String (read/write)	Display name of the device
Parent	String (read)	OID of the parent object (for subdevices only)
Power	String (read)	OID of the parent object (for subdevices only)
Software_version	String (read)	Version of the software currently installed on the device
State Onoff		Device power state
Status	String (read)	Active, inactive, installing, or disabled
Enable	Boolean (read/write)	False when the device is intentionally disabled
Energy	Float (k Wh) (read)	The amount of energy that is used by the device
Hardware version	String (read)	Version of the hardware

`install` Interface

The `install` interface has the following methods:

`install`. Implement the `install` method to make a device fetch and install a new software image from the supplied URL.

`attributes`. The `attributes` method uses a URL to point to the location of the software to be installed on the device.

`admin` Interface

The `admin` interface has the following methods:

`reboot`. The `reboot` method changes the `status` attribute to `rebooting` until the smart home device finishes rebooting

`reset`. The `reset` method resets the device's configuration to factory defaults. This might cause the device to also reboot. This method changes the device `status` attribute to `resetting` until the configuration is completely reset. The `A` and `B` attributes offer some protection against accidental configuration resets caused by buggy software.

- `A`. The value has to be 08914260 for the reset to occur.
- `B`. The value has to be 28912070 for the reset to occur.

`switch` Extended Type

The `switch` type controls the power of the device with its `state` attribute. Values can be `on` or `off`.

`dimmer` Extended Type

The `dimmer` type controls the brightness of a lamp. This type has the following attributes:

`state`. A power setting value of either `on` or `off`.

`brightness`. The level of brightness from 0 to 100 (0 = full-off, 100 = full-on).

Setting `state` to `on` turns the light on to the previously set brightness level. Setting `state` to `off` turns the light off completely.

`thermostat` Extended Type

The `thermostat` type senses temperature and controls heating and cooling. This type has the following attributes:

`temperature`. The current temperature.

`setpoint`. The desired temperature.

`mode`. The heating and cooling mode. Values can be `off`, `auto`, `heat`, or `cool`.

binary_sensor Extended Type

The binary_sensor type monitors events and tracks people or objects as they move through an area.This type has the following attributes:

output. 0 or 1.

enable. A Boolean value.

motion_sensor Extended Type

The motion_sensor type extends the binary_sensor type and tracks people or objects as they move through an area. This type has the following attributes:

reset_period. The time in seconds that the output stays true after the sensor detects motion.

threshold. A value from 0 to 100 that sets the sensitivity threshold. A lower threshold sets it to a higher sensitivity).

sound_sensor Extended Type

The sound_sensor type extends the binary_sensor type. This type has the following attributes:

reset_period. The time in seconds output stays true after the sensor detects motion.

threshold. A value from 0 to 100 (lower threshold means higher sensitivity).

contact_sensor Extended Type

The contact_sensor type extends the binary_sensor type. A contact sensor detects contact closure.

shade Extended Type

The shades type controls window shades. This type has the following attributes:

position. A value of either open or closed. Setting position to open opens the shade to the previously set level. Setting position to closed closes the shade completely.

level. A value that determines how much of the window is covered by the shade. A value of 0 means the shade is fully extended (closed). A value of 100 means the shade is fully retracted (open). At level 0 the shade is fully extended and closed. The light allowed in is reduced to 0 so that 0% of the window is exposed. At level 100 the shade is fully retracted and open. The light allowed in is 100% so that 100% of the window is exposed.

angle. A value that specifies the number of degrees at which the shade slats are angled.

`ramp` Interface

A class of type `shade` can implement the `ramp` interface to adjust the level of the shade. This type has the following attributes:

`ramping`. The current `ramp` state. Values can be `up`, `down`, or `stop`.

`ramp_up`. Open shade.

`ramp_down`. Close shade.

`ramp_stop`. Stop opening or closing (as applicable).

`scene_controller` Extended Type

The `scene_controller` type supports up to N buttons, each of which can be programmed to run a specific scene. This type has the `scene_N` attribute, which is a FLUID object ID of the scene to run when button N is pressed. Leave this attribute empty if there is no scene.

Object Examples

Create a New Object

Request from `cleint1@controlpoint1` that `client2@controlpoint2` create an object named `light12` with an brightness of `80`.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <new type="light12" brightness="80"/>
</fcml>
```

Response from `client2@controlpoint2` that it has a new object 7 (`controlpoint1.client2.object7`) named `foo` with an attribute of `bar`.

```
<fcml from="c2@d1">
   <controlpoint1.client2.object7.is_new type="light12" brightness="80"/>
</fcml>
```

Delete an Object

Request from `client1@controlpoint1` that `client2@controlpoint1` delete its object 7 (`object7`).

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <object7.delete/>
</fcml>
```

Response from `client2@controlpoint2` that its `object7` is deleted.

```
<fcml from="client2@controlpoint2">
```

```
   <controlpoint1.client2.object7.is_deleted/>
</fcml>
```

Error Deleting an Object

Request from `client1@controlpoint1` that `client2@controlpoint1` delete its `object7`.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <object7.delete/>
</fcml>
```

Response from `client2@controlpoint1` with error code `1`, which means it could not delete `object7` because `object7` does not exist.

```
<fcml from="client2@controlpoint1" to="client1@controlpoint1">
   <error description="Object does not exist" code="1">
      <object7.delete/>
   </error>
</fcml>
```

Attribute Examples

Get Attribute Information

Request from `client1@controlpoint1`) to `client2@controlpoint1`) to retrieve (`get`) information from `light1` on `client2` and return that information to `client1`.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <light1.get/>
</fcml>
```

Response from `client2@controlpoint1` that `controlpoint1.client2.light1` is type `light` with attributes `brightness` and `state`.

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.light1.is type="light" brightness="80" state=”on”/>
</fcml>
```

Add an Attribute

Request from `client1@controlpoint1` that `client2@controlpoint1` set the `brightness` attribute on `light1` to `80`.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <light1.set brightness="80"/>
</fcml>
```

Response from client2@controlpoint1 that the attribute brightness on controlpoint1.client2.object1 is set to 80.

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.object1.is brightness="80"/>
</fcml>
```

Modify an Attribute

Request from client1@controlpoint1 that client2@controlpoint1 change attribute b on object1 from puppy to dog.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <object1.set b="dog"/>
</fcml>
```

Response from client2@controlpoint1 that attribute b on its object1 is dog.

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.object1.is b="dog"/>
</fcml>
```

Delete an Attribute

Request from client1@controlpoint1 that client2@controlpoint1 delete its attribute b.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <o1.delete attributes="b"/>
</fcml>
```

Response from client2@controlpoint1 that its attribute b is deleted.

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.object1.is_deleted attributes="b"/>
</fcml>
```

Send Transitory Information (says Method)

Request from client2@controlpont1 that client1@controlpoint1 return password information about o1. In this example, the information returned is that the password on object1 has been changed.

```
<fcml from="client2@controlpoint1" to="client1@controlpoint1">
   <controlpoint1.client2.object1.says key="password_changed"
                        description="The password has been changed."/>
</fcml>
```

Element Examples

Add an Element

Request from `client1@controlpoint1` that `client2@controlpoint1` add an element to its `object1` and set its attributes a and b to `bark` and `wag`. The element then sleeps until it receives a request to wake up, at which point it barks and wags.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <object1.set>
      <e a="bark" b="wag">
         <f>sleep</f>
      </e>
   <o1.set/>
</fcml>
```

Response from `client2@controlpoint1` that `controlpoint1.client2.object1` has a sleeping element (`e1`) that barks and wags when it wakes up.

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.object1.is>
      <e id="e1" a="bark" b="wag">
         <f>sleep</f>
      </e>
   <controlpoint1.client2.object1.is/>
</fcml>
```

Modify an Element

Request from `client1@controlpoint1` that `client2@controlpoint1` change attribute a on element 1 (`e1`) on `o1` to `woof` instead of `bark` and that `e1` wake up (`wake`).

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <object1.set>
      <e id="e1" a="woof">
         <g>wake</g>
      </e>
   <object1.set/>
</fcml>
```

Response from `client2@controlpoint1` that attribute a on `e1` is set to `woof`, and that `e1` is awake (`wake`).

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.object1.is>
      <e id="e1" a="woof">
```

```
         <g>wake</g>
      </e>
   <controlpoint1.client2.object1.is/>
</fcml>
```

Delete an Element

Request for `client1@controlpoint1` that `client2@controlpoint1` delete its `e1`.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <object1.delete elements="e1"/>
</fcml>
```

Response from `client2@controlpoint1` that its `e1` is deleted.

```
<fcml from="client2@controlpoint1">
   <controlpoint1.client2.object1.is_deleted elements="e1"/>
</fcml>
```

Search for Object Information

Request from `client1@controlpoint1` that `client2@controlpoint1` search for objects that contain `light` and return a result object (`_result_d1_c1_1`) with the first 3 matches found.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <search path="light" result="_result_controlpoint1_client1_1" limit="3"/>
</fcml>
```

Response from `c2@d1` that the search resulted in finding 6 objects matching the search criteria, and the first 3 found objects are in the first response.

```
<fcml from="client2@controlpoint1" to="client1@controlpoint1" >
   <_result_controlpoint1_client1_1.is_new type="result" path="light" limit="3"
                                          start="1" count="6">
      <ref>object1</ref>
      <ref>object2</ref>
      <ref>object3</ref>
   </_result_controlpoint1_client1_1.is_new>
</fcml>
```

Get More Search Results

Request from `client1@controlpoint1` that `client2@controlpoint1` return 3 results of the search described in *Search for Object Information* on page 59 starting with result 4.

```
<fcml from="client1@controlpoint1" to="client2@controlpoint1">
   <_result_controlpoint1_client1_1.set start="4"/>
</fcml>
```

Response from `c2@d1` with the next 3 search results starting with result 4.

```
<fcml to="client1@controlpoint1" from="client2@controlpoint1">
   <_result_d1_c1_1.is start="4">
      <ref>o4</ref>
      <ref>o5</ref>
      <ref>o6</ref>
   </_result_controlpoint1_client1_1.is>
</fcml>
```

Router Management APIs

5

control points and network routing

This chapter explains the FCML messages an control point uses for network router management. Use the information in this chapter to add FCML communications to client plug-ins that run an control point.

This chapter covers the following topics:

Authentication

The `netrouter` client that runs on the control point manages all interactions with network routing functionality. The `netrouter` client requires authentication before a client application can do things such as query or change configuration information or troubleshoot network routing issues.

Each new session is issued a `sessionId`. The number of simultaneous sessions is restricted to avoid conflicts. The `sessionId` holder must terminate the session as soon as possible with a call to `SessionManager.endSession` so that other services can access the netrouter resources. Sessions that are not terminated time out after a period of inactivity, which is currently 10 minutes.

The `netrouter` client requires authentication in the following format:

Request

```
<fcml to="netrouter@cpX" from="a@b">
   <SessionManagement.startSession username="myUserName"
                                          password="myPassword"/>
</fcml>
```

Response

The `sessionID` returned in the response is then used on all subsequent calls to the `netrouter` client.

```
<fcml to="a@b" from="netrouter@cpX">
   <SessionManagement.startSession sessionID="1743d38c47627b68"/>
</fcml>
```

End Session

```
<fcml to netrouter@cpX from a@b>
   <SessionManagement.endSession sessionId="1743d38c47627b68">
</fcml>
```

Syntax

You form a network router management FCML request by concatenating the object and method name with a period followed by the attributes:

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.ConfigurationStarted="1743d38c47627b68" />
</fcml>
```

Response

A response is formed the same way and includes a response code as follows:

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.ConfigurationStarted Ihop="1743d38c47627b68"
                                                    ResponseCode="1" />
</fcml>
```

`GetInfo` Methods

The device configuration objects that run on the `netrouter` client do not follow the standard FCML convention that allows `is` and `get` calls. Instead, a `GetInfo` method is defined for each device configuration object that requests the information as follows:

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.GetInfo _sessionId="1743d38c47627b68" />
</fcml>
```

The response returns the information in the `GetInfo` method attributes and `ResponseCode` as follows:

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.GetInfo _sessionId="1743d38c47627b68" BlankState="0"
            NewBlockSiteEnable="0" NewBlockSiteName="A Name" NewTimeZone="11"
            NewDaylightSaving="0" ResponseCode=”1”/>
</fcml>
```

Object Response Codes

Every device management object has the same `ResponseCode` attribute that returns one of the values that are listed here depending on the outcome of the request:

Attributes	Value	Description
ResponseCode	0	No error.
	1	Reboot required.
	401	Invalid action.
	402	Invalid arguments.
	501	Action failed.
	605	String argument too long.
	702	Value that is specified is invalid.
	728	Invalid channel.

`DeviceConfig` Object

The `DeviceConfig` object has methods for starting and finishing a session, and for managing traffic, firmware updates, reboots, factory default resets, and time zone information.

`ConfigurationStarted` Method

Attributes	Data Type	Description
_sessionId	String	The ID for the session up to 35 characters.

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.ConfigurationStarted _sessionId="1743d38c47627b68" />
</fcml>
```

Response

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.ConfigurationStarted _sessionId="1743d38c47627b68"
                                      ResponseCode="1" />
</fcml>
```

ConfigurationFinished Method

Attributes	Value	Description
NewStatus	ChangesApplied RebootRequired	The requested changes are applied. Some of the requested changes might require a router reboot.

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.ConfigurationFinished _sessionId="1743d38c47627b68"
                                       NewStatus="ChangesApplied" />
</fcml>
```

Response

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.ConfigurationFinished _sessionId="1743d38c47627b68"
                                       ResponseCode="1" />
</fcml>
```

Reboot Method

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.Reboot _sessionId="1743d38c47627b68" />
</fcml>
```

Response

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.Reboot _sessionId="1743d38c47627b68" ResponseCode=”1” />
</fcml>
```

SetTimeZone Method

Attributes	Value	Description
NewTimeZone	–12 to +12	Minus or plus up to 12 hours.
NewDaylightSaving	1	Enabled.

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.SetTimeZone _sessionId="1743d38c47627b68"
                             NewTimeZone="11" NewDaylightSaving="0"/>
</fcml>
```

Response

```
<fcml to="x@y" from="netrouter@cp2">
      <DeviceConfig.SetTimeZone _sessionId="1743d38c47627b68"
                                NewTimeZone="11" NewDaylightSaving="0"/>
</fcml>
```

GetTimeZoneInfo Method

Attributes	Value	Description
NewTimeZone	-12 to +12	Minus or plus 12 hours.
NewDaylightSaving	1	Enabled.

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.GetTimeZoneInfo _sessionId="1743d38c47627b68" />
</fcml>
```

Response

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.GetTimeZoneInfo _sessionId="1743d38c47627b68"
            NewTimeZone="11" NewDaylightSaving="0" ResponseCode=”1” //>
</fcml>
```

GetInfo Method

Attributes	Value	Description
Blank State	0 1	Disabled. Enabled.
NewBlockSiteEnable	0 1	Disabled. Enabled.
NewBlockSiteName	0 1	Disabled. Enabled.
NewTimeZone	0 1	Disabled. Enabled.
NewDaylightSaving	0 1	Disabled. Enabled.

EnableTrafficMeter Method

Attributes	Value	Description
NewTrafficMeterEnable	0 1	Disabled. Enabled.

GetTrafficMeterEnabled Method

Attributes	Value	Description
NewTrafficMeterEnable	0	Disabled.
	1	Enabled.
	2	Traffic meter service not supported.

Request

```
<fcml to="netrouter@cp2" from="x@y">
   <DeviceConfig.GetInfo _sessionId="1743d38c47627b68" />
</fcml>
```

Response

```
<fcml to="x@y" from="netrouter@cp2">
   <DeviceConfig.GetInfo _sessionId="1743d38c47627b68" BlankState="0"
```

```
            NewBlockSiteEnable="0" NewBlockSiteName="A Name" NewTimeZone="11"
            NewDaylightSaving="0" ResponseCode=”1”/>
</fcml>
```

SetTrafficMeterOptions Method

Attributes	Data Type	Description
NewControlOption	String	Values can be: No Limit. Download only. Both Directions.
NewMonthlyLimit	Integer	The traffic limit up to 1 million Mbps.
RestartHour	Integer	0–24.
RestartMinute	Integer	0–60.
RestartDay	Integer	1–31.

GetTrafficMeterOptions Method

Attributes	Data Type	Description
NewControlOption	String	Values can be: No Limit. Download only. Both Directions.
NewMonthlyLimit	Integer	The traffic limit up to 1 million Mbps.
RestartHour	Integer	0–24.
RestartMinute	Integer	0–60.
RestartDay	Integer	1–31.

GetTrafficMeterStatistics Method

Attributes	Data Type	Description
NewTodayConnectionTime	String	The connection time in hh:mm format.
NewTodayUpload	Double or String	Uploads today in Mbps.
NewTodayDownload	Double or String	Downloads today in Mbps.
NewYesterdayConnectionTime	String	The time in hh:mm format.
NewYesterdayUpload	Double or String	Uploads yesterday in Mbps.
NewYesterdayDownload	Double or String	Downloads yesterday in Mbps.
NewWeekConnectionTime	String	The time in hh:mm format.
NewWeekUpload	Double or String	Uploads this week in Mbps.
NewWeekDownload	Double or String	Downloads this week in Mbps.
NewMonthConnectionTime	String	The time in hh:mm format.
NewMonthUpload	Double or String	Uploads this month in Mbps.
New Month Download	Double or String	Downloads this month in Mbps.
NewLastMonthConnectionTime	String	The time in hh:mm format.
NewLastMonthUpload	Double or String	Uploads last month in Mbps.
NewLastMonthDownload	Double or String	Downloads last month in Mbps.

CheckNewFirmware Method

Attributes	Data Type	Description
CurrentVersion	String	The current firmware version.
NewVersion	String	The new firmware version.
ReleaseNote	String	Release notes up to 512 characters.

UpdateNewFirmware Method

Attributes	Value	Description
YesOrNo	0 1	Cancel. Update.

ResetToFactoryDefault Method

Attributes	Data Type	Description
RouterPassword	String	The router password up to 32 characters.
SerialNumber	String	The router serial number up to 32 characters.

DeviceInfo Object

The DeviceInfo object returns device information and system information. The returned information includes system uptime and attached devices.

GetInfo Method

Attributes	Data Type	Examples
ModelName	String	WNR834Bv2.
Description	String	NETGEAR Smart Wizard 3.0, specification 0.8 version.
SerialNumber	String	1ML1747D0000B.
Firmwareversion	String	V2.0.1.
SmartAgentversion	String	3.0.
FirewallVersion	String	ACOS NAT/FW v2.5.0.44 (Linux Cone NAT Hot Patch 09/14/2007).
VPNVersion	String	N/A.
OthersoftwareVersion	String	2.0.1NA.
Hardwareversion	String	WNR834A.
Otherhardwareversion	String	N/A.
FirstUseDate	String	Sunday, 20 Sep 2007 01:10:03.

GetSysUpTime Method

Attributes	Format	Description
SysUpTime	HH:MM:SS	The cumulative time in hours, minutes, and seconds. For example, if the system is up for 1.5 days, the value would have this format: 36:xx:xx.

GetAttachDevice Method

Attribute	Format	Description	
NewAttachDevice	n@Index;IP;Name;MAC;type;linkspeed@…	n	Count of attached devices.
		Index	Index of each device.
		IP	IP address of each device.
		Name	Name of each device.
		MAC	MAC address of each device.
		Type	The client device connection type: wired or wireless.
		Linkspeed	The link rate of each client in Mbps (PHY rate): Wireless client: Up to 300. Fast Ethernet client: 100. Gigabit client: 1000.

Example:

002@1;192.168.1.2;Peter'sNB;11:22:33:44:55:66;wired;100@2;192.168.1.3;William'sNB;11:22:33:44:55:77;wireless;3

WANIPConnection Object

The WANIPConnection object has methods for managing port settings and connection type settings. Also included are ISP, DSL, and DNS security settings, and other security settings such as MAC addressing and port mapping. Smart Wizard detection is also included.

SetConnectionType Method

Attributes	Data Type/Value	Description
NewConnectionType	DHCP Static PPPoE PPPoA PPTP	Dynamic Host Configuration Protocol. Fixed IP address. Point-to-point over Ethernet. Point-to-point over ATM. Point-to-point Tunneling Protocol.
NewISPLoginname	String	The new ISP login name up to 64 characters.
NewISPPassword	String	The new ISP password up to 64 characters.

Attributes	Data Type/Value	Description
NewConnectionMode	AlwaysOn DialOnDemand ManuallyConnect N/A (for DHCP, static)	The mode that tells the router to manually or automatically reconnect to the Internet after idling.
Newidletimer	Integer	The idle time up to 99 seconds.
NewConnectionID	String	The new connection ID up to 12 characters (for PPTP).
NewPPTPMyIP	IP, PPoA	An address in the format: xxx.xxx.xxx.xxx.
NewPPTPServerIP	IP, PPPoA	An address in the format: xxx.xxx.xxx.xxx.
NewBigpoindAuthServer	IP, PPPoA	An address in the format: xxx.xxx.xxx.xxx.
NewPrimaryDNS	IP, PPPoA	An address in the format: xxx.xxx.xxx.xxx A value of 0.0.0. means Get Automated DNS from ISP.
NewSecondaryDNS	IP, PPoA	An IP address in the format: xxx.xxx.xxx.xxx.
NewNATEnable	0 1 2	Off. On. Disable firewall.

SetConnection2Type Method

Attributes	Data Type/Value	Description
NewISP2Loginname	String	The new ISP login name up to 32 characters.
NewISP2Password	String	The new IPS password up to 16 characters.
NewNAT2Enable	0 1 2	Off. On. Disable firewall.

SetDSLConfig Method

Attributes	Data Type	Description
NewWANEncap	String	Values can be: LLC-Based. VC-BASED.
NewWANVCI	Integer	3 -102.
NewWANVPI	Integer	0-10.

SetIPInterfaceInfo Method

Attributes	Data Type/Value	Description
NewAddressingType	String	Value is Static (only).
NewExternalIPAddress	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewSubnetMask	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewDefaultGateway	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewPrimaryDNS	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewSecondaryDNS	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.

SetMACAddress Method

Attributes	Value	Description
NewMACAddress	AABBCCDDEEFF	The MAC address of the device.
NewMACAddressOverride	0 1 2	Use default MAC address. MAC address of the customer computer. Customer-specified MAC address.

NewSmartWizardDetection Method

Attributes	Value	Description
NewConnectionType	Down DHCP Static PPPoE PPPoA PPTP	The connection is down. Dynamic Host Configuration Protocol. Fixed IP address. Point-to-Point over Ethernet. Point-to-Point over ATM. Point-to-Point Tunneling Protocol.

AddPortMapping Method

Attributes	Data Type/Value	Description
NewProtocol	TCP UDP	Transmission Control Protocol (TCP). User Datagram Protocol (UDP).
NewInternalPort	Integer	The number of the internal port that is the gateway to every device within the internal home network. Values are 1–65534.
NewExternalPort	Integer	The number of the external port that is accessible to devices outside the home network. Values are 1–65534.
NewInternalClient	String	The IP address of the internal device in the format: xxx.xxx.xxx.xxx.
NewPortMappingDescription	String	Length <= 64 characters.
NewPortMappingAdd	1	Set value to 1 to add port mapping.

DeletePortMapping Method

Attributes	Data Types/Value	
NewProtocol	TCP UDP	Transmission Control Protocol (TCP). User Datagram Protocol (UDP).
NewExternalPort	Integer	The number of the external port that is accessible to devices outside the home network. Values are 1-65534.
NewPortMappingAdd	0	Set value to 0 to delete this mapping.

GetConnectionTypeInfo Method

Attributes	Value	
NewConnectionType	Down DHCP Static PPPoE PPPoA PPTP	The connection is down. Dynamic Host Configuration Protocol. Fixed IP address. Point-to-Point over Ethernet. Point-to-Point over ATM. Point-to-Point Tunneling Protocol.

GetPPPConnStatus Method

Attributes	Data Type	
NewConnectionTime	String	The new connection time.
NewConnectStatus	0 1	Connected. Waiting for data.
NewNegotiationStatus	0 1 2 3 4 -1	Success Received short protocol reject packet. LCP down. LCP is allowed to come up. Initialize LCP. Unknown error.
NewAuthenticationStatus	0 1 2 -1	Success. CHAP authentication failed. PAP authentication failed. Unknown error.

GetModemInfo Method

Attributes	Data Type/Value	Description
NewModemVersion	String	The new modem version.
NewModemStatus	0 1 2 -1	Connected. Negotiating. Connecting. Unknown error.

Attributes	Data Type/Value	Description
NewDownStreamSpeed	String	The new downstream speed.
NewUpStreamSpeed	String	The new upstream speed.
NewWANVCI	Integer	32–102.
NewWANVPI	Integer	0–10.

GetInfo Method

Attributes	Data Type/Value	Description
NewEnable	0 1	Disabled. Enabled.
NewConnectionType	Down DHCP Static PPPoE PPPoA PPTP	The connection is down. Dynamic Host Configuration Protocol. Fixed IP address. Point-to-Point over Ethernet. Point-to-Point over ATM. Point-to-Point tunneling protocol.
NewExternalIPAddress	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewSubnetMask	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewAddressingType	Down DHCP Static PPPoE PPPoA PPTP	The connection is down. Dynamic Host Configuration Protocol. Fixed IP address. Point-to-point over ethernet. Point-to-point over ATM. Point-to-point tunneling protocol.
NewDefaultGateway	IP, IPoA	An address in the format: xxx.xxx.xxx.xxx.
NewMACAddress	AABBCCDDEEFF	The MAC address of the device.
NewMACAddressOverride	0 1	Disabled. Enabled.
NewMaxMTUSize	Integer	The new maximum MTU size up to 1500 Mbps.
NewDNSEnabled	0 1	Disabled. Enabled.
NewDNSServers	IP	An address in the format: xxx.xxx.xxx.xxx.

GetPortMappingInfo Method

Attributes	Value	Description
NewPortMappingNumberofEntries	1-10	The number of port mapping entries.
NewPortMappingInfo	Protocol;internal_port; external_port;client_IP; description@ Protocol; internal_port; external_port; client_IP; description protocol is "TCP" or "UDP" internal_port is "Internal Port" external_port is "External Port" client_IP is "Client site IP address" description is "Description about this port mapping entry"	The protocol in use with a description. The protocol and description are delimited by the at sign (@).

WLANConfiguration Object

The WLANConfiguration object provides a method for managing the WLAN configuration including security settings.

SetEnable Method

Attributes	Value	Description
NewEnable	0 1	Disabled. Enabled.

SetConfigPassword Method

Attributes	Data Type	Description
NewPassword	String	The new password up to 64 characters.

SetChannel Method

Attributes	Data Type	Description
NewChannel	Digits	Auto. 1–13. 1–11 (U.S. only). 6 when Dynamic Super G.

Set5GChannel Method

Attributes	Data Type	Description
New5GChannel	Digits	Auto 36 40 44 48 52 56 60 64

SetSSID Method

Attributes	Data Type	Description
NewSSID	String	The network name up to 32 characters.

SetSSIDBroadcast Method

Attributes	Value	Description
NewSSIDBroadcast	0 1	Disabled. Enabled.

Set5GSSID Method

Attributes	Data Type	Description
New5GSSID	String	The 5G network name up to 32 characters.

SetWPSMode Method

Attributes	Value	Description
NewWPSEnable	0 1	Disabled. Enabled.

PressWPSPBC Method

The PressWPSPB method simulates the pressing of the physical WPS push button to enable the WPS process.

GetInfo Method

Attributes	Data Type/Value	Description
NewEnable	0 1	Disabled. Enabled.
NewSSIDBroadcast	0 1	Off. On.
NewStatus	Up Disable	This link status.
NewSSID	String	The network name.
NewRegion	String	The region value. See *Region Values* on page 158.
NewChannel	Digits	Auto. 1–13. 6 for Dynamic Super G. 1–11 for U.S. only.

Attributes	Data Type/Value	Description
NewWirelessMode	String	Values can be the following: b. g. Auto 108 Mbps. g & b. 130 Mbps. 270 Mbps.
NewBasicEncryptionModes	None WEP WPA-PSK WPA2-PSK WPA-PSK/WPA2-PSK	The type of data encryption in effect. WPA-PSK/WPA2-PSK is recommended.
NewWEPAuthType	Open Shared Automatic	The type of authentication in effect.
NewWPAEncryptionModes	None WPA-PSK WPA2-PSK WPA-PSK/WPA2-PSK	The type of data encryption in effect. WPA-PSK/WPA2-PSK is recommended.
NewWLANMACAddress	AABBCCDDEEFF	A unique identifier that is assigned to the device.

GetWEPSecurityKeys Method

Attributes	Data Type	Description
NewWEPKey	String	The new WEP encryption key that can be 10 or 26 characters.
NewWEPPassphrase	String	The new WEP passphrase.

GetWPASecurityKeys Method

Attributes	Data Type	Description
NewWPAPassphrase	String	The new WPA passphrase.

GetSSID Method

Attributes	Data Type	Description
NewSSID	String	The new network name.

Get5GSSID Method

Attributes	Data Type	Description
New5GSSID	String	The new 5G network name.

GetChannelInfo Method

Attributes	Value	Description
NewChannel	Auto 1–3 6 for Dynamic Super G 0qP11 U.S. only	The new wireless channel number.

Get5GChannelInfo Method

Attributes	Value	Description
New5GChannel	Auto 36 40 44 48	The new 5G wireless channel number.

Get5GInfo Method

Attributes	Value	Description
New5GSSID	String	The new 5G network name.
New5GChannel	Auto 36 40 44 48	The new 5G wireless channel number.

GetRegion Method

Attributes	Data Type	Description
NewRegion	String	The new region value up to 64 characters.

GetWirelessMode Method

Attributes	Data Type	Description
NewWirelessMode	String	Values can be the following: b. g. Auto 108 Mbps. g & b. 130 Mbps. 270 Mbps.

GetSSIDBroadcast Method

Attributes	Value	Description
NewSSIDBroadcast	0 1	Off. On.

GetWPSMode Method

Attributes	Value	Description
NewWPSEnable	0 1	Disabled. Enabled.

GetWPSPINInfo Method

Attributes	Data Type/Value	Description
NewWPSPINEnable	0 1	Disabled. Enabled.
NewRouterWPSPIN	String	The WPS PIN number.

Dual-Band Router WLANConfiguration Methods

The four methods that are described in this section can be used with dual-band routers that support both of the 2.4 GHz and 5 GHz bands. When you call any of the four methods, apply the same settings to both bands except for the NewSSID and NewChannel attributes:

NewSSID. This attribute applies only to 2.4 GHz. The 5 GHz SSID is applied through the set5GSSID(New5GSSID) API instead.

NewChannel. This attribute applies only to 2.4 GHz. Set this attribute to Auto or 44. If Auto is not supported, 44 is preferred over 36. This preference is because 5 GHz internally has a higher transmission output power in the mid-channel than in the band-edge channel.

The following four methods have a new value for the NewWirelessMode attribute. The new value is 270 Mbps_5G. This value tells the router to do two things: 1) Configure the 5 GHz band to the 270 Mbps rate, and 2) Set the 2.4 GHz band to allow 270 Mbps as its maximum rate. For example, the WNDR3300 allows 54 Mbps for 2.4 GHz when 5 GHz is set to 270 Mbps.

- SetWLANNoSecurity method
- SetWLANWEPByKeys method
- SetWLANWEPByPassphrase method
- SetWLANWPAPSKByPassphrase method

SetWLANNoSecurity *Method*

Attributes	Data Type	Description
NewSSID	String	The new network name up to 32 characters..
NewRegion	String	The new region value up to 64 characters. See *Region Values* on page 158.
NewChannel	Digits	Auto. 1–13. 6 for Dynamic Super G. 1–11 U.s. only.
NewWirelessMode	String	Values can be the following: b. g. Auto 108 Mbps. g & b. 130 Mbps. 270Mbps. 270 Mbps_5G (for dual band).

SetWLANWEPByKeys *Method*

Attributes	Data Type/Value	Description
NewSSID	String	The network name up to 32 characters.
NewRegion	String	The region value up to 64 characters. See *Region Values* on page 158.
NewChannel	Digits	Auto. 1–13. 6 for Dynamic Super G. 1–11 U.S. only.

Attributes	Data Type/Value	Description
NewWirelessMode	String	Values can be the following: b. g. Auto 108 Mbps. g & b. 130 Mbps. 270 Mbps. 270 Mbps_5G (for dual band).
NewWEPAuthType	Automatic Open Shared	The type of WEP authentication.
NewWEPLength	String	The new WEP key bit length that can be 64 or 128 characters.
NewKey1	String	The new WEP encryption key that can be 10 or 26 characters.
NewKey2	String	The second new WEP encryption key that can be 10 or 26 characters.
NewKey3	String	The third new WEP encryption key that can be 10 or 26 characters.
NewKey4	String	The fourth new WEP encryption key that can be 10 or 26 characters.

SetWLANWEPByPassphrase *Method*

Attributes	Data Type/Value	Description
NewSSID	String	The new network name up to 32 characters.
NewRegion	String	The new region value that can be up to 64 characters. See *Region Values* on page 158.
NewChannel	Digits	Auto. 1–13. 6 for Dynamic Super G. 1–11 U.S. only.

Attributes	Data Type/Value	Description
NewWirelessMode	String	Values can be the following: b. g. Auto 108 Mbps. g & b. 130 Mbps. 270 Mbps. 270 Mbps_5G (for dual band).
NewWEPAuthType	Automatic Open Shared	The new type of WEP authentication.
NewWEPLength	String	The new WEP bit length that can be 64 or 128 characters.
NewWEPPassphrase	String	The new WEP passphrase up to 32 characters.

SetWLANWPAPSKByPassphrase *Method*

:

Attributes	Data Type	Description
NewSSID	String	The network name up to 32 characters.
NewRegion	String	The new region value up to 64 characters. See *Region Values* on page 158.
NewChannel	Digits	Auto. 1-13. 6 for Dynamic Super G. 1-11 U.s. only.
NewWirelessMode	String	Values can be the following: b. g. Auto 108 Mbps. g & b. 130 Mbps. 270 Mbps. 270 Mbps_5G (for dual band).

Attributes	Data Type	Description
NewWPAEncryptionModes	WPA-PSK/WPA2-PSK WPA-PSK WPA2-PSK	The WPA encryption mode.
NewWPAPassphrase	String	The new WPA passphrase that can be from 8 through 64 characters.

ParentalControl Object

The ParentalControl object provides methods for filtering internet access to keep unwanted content out of your network.

GetDNSMasqDeviceID Method

Attributes	Data Type	Description
NewMACAddress	String	The MAC address of the device.

Authenticate Method

Attributes	Data Type	Description
NewUsername	String	The new user name up to 64 characters.
NewPassword	String	The new password up to 64 characters.

SetDNSMasqDeviceID Method

Attributes	Data Type	Description
NewMACAddress	String	The new MAC address that is 16 characters.
NewDevicdID	String	The new device ID that is 32 characters.

EnableParentalControl Method

Attributes	Value	Description
NewEnable	0 1	Disabled. Enabled.

GetAllMACAddresses Method

Responses:

Attributes	Data Type	Description
AllMACAddresses	String	All MAC addresses delimited by | up to 1024 characters.

DeleteMACAddress Method

Request:

Attributes	Data Type	Description
NewMACAddress	String	The new MAC address that is 16 characters.

GetEnableStatus Method

Attributes	Value	Description
ParentalControl	0 1	Disabled Enabled

Control Point APIs

6

Portal server to control point communications

This chapter explains the FCML message the portal server applications use to communicate with an control point. It also explains how control point clients communicate with each other. Use the information in this chapter to add FCML communications to your client plug-ins.

This chapter covers the following topics:

Control Point Clients

When a portal application communicates with a control point, it addresses an FCML message request to the applicable control point client. Each control point client uses the optional `cp` domain to communicate with each other within the same control point.

`router` Client

The `router` client is the core diagnostic engine. This client implements the following methods:

- **`snapshot` method**. The portal application calls this method to get a listing of the current attribute settings in the `router` client.
- **`deactivate` method**. The portal application calls this method to deactivate the `router` client.
- **`restart` method**. The portal application calls this method to restart the `router` client.
- **`reset` method**. The portal application calls this method to reset the `router` client.
- **`upload` method**. The portal application calls this method to upload bundles to the `router` client.
- **`disconnect` `method`**. The portal application calls this method to disconnect the `router` client.

`_ifconfig` Object

The `_ifconfig` object is type `System` and has the `friendly_name` attribute of type `String`, and the `CDATA` element. See *Elements* on page 42 for a description of the `CDATA` element.

_config Object

The _config object is type System and has the following attributes.

Attributes	Data Type	Description
https_enabled	Boolean	A flag that describes whether HTTP or HTTPS is in use.
http_port	Integer	The port number for HTTP. Normally this number is 80.
https_port	Integer	The port number for HTTPS. Normally this number is 443.

_lan Object

The _lan object is type LanSystemObject and has the following attributes.

Attributes	Data Type	Description
ip_address	String	The IP address of the control point.
subnet_mask	String	The subnet mask of the control point.
default_gateway	String	The IP address of the default gateway.
dns_1	String	The IP address of the first DNS server.
dns_2	String	The IP address of the second DNS server.

_messages Object

The messages object is type MessageSystemObject. This object has the url attribute of type String.

netrouter Client

The netrouter client is the router snapshot capture plug-in. This client is the core diagnostic engine. This client implements the following methods:

- **snapshot method**. The portal application calls this method to get a listing of the current attribute settings in the netrouter client.
- **startSession method**. The portal application calls this method to start a netrouter session.
- **endSession method**. The portal application calls this method to end a netrouter session.
- **reset method**. The portal application calls this method to reset a netrouter session.
- **configurationStarted method**. The portal application calls this method to start a netrouter client configuration.

DeviceConfig Object

The DeviceConfig object is type String and has the sessionID attribute of type String.

device Client

The devices client is the watching service for new devices. This client has the following attributes, and implements the search method. The search method searches for object information and is described in *Search for Object Information* on page 59.

This client stores instances of the following types: dimmer, switch, thermostat, camera, binary_sensor, motion_sensor, keypad, shade, fan, scene_controller, camera_motion_sensor, sound_sensor, meter, door_lock.

upnp Client

The upnp client is the UPnP device monitor plug-in. Objects of type upnp have the following attributes.

Attributes	Data Type	Description
manufacturer	String	The name of the UPnP manufacturer
model_name	String	The model name of the UPnP device
device_type	String	The type of the UPnP device
address	String	The IP address of the UPnP device
content	String	The URL to the content on the UPnP device.
active	String	Whether the UPnP device is active on the network.
discovered	String	Whether this UPnP device was discovered by the UPnP client.
type	String	The type of the UPnP device. For example, camera, motion sensor, and so on.
uuid	String	Universal Unique ID of the UPnP device.
model_number	String	The model number of the UPnP device.

trig Client

The trig client accepts trigger objects and executes them when their conditions become true. In this example, the trigger object contains a command to send an _ifconfig.get message to router@cp3 each day at 3:15 am.

```
<fcml from="monitoring@portal" to="vns">
   <obj1.new type="event" trigger="(tss.time.time=='03:15')"
         activate="0" name="triggerName">
      <fcml to="router@cp3" reply_to="monitoring@portal">
         <_ifconfig.get/>
      </fcml>
   </obj1.new>
</fcml>
```

tss Client

The tss client is the temporal synchronization service. This client announces the time once per minute.

_config Object

The _config object has the time_zone attribute of type String.

vns Client

The vns client is the virtual node service. VNS is a local database for FCML objects. The VNS database can contain any type of object. This client implements the following methods.

new method. This method is described on page 49. When the new method is called, it creates a new object of the given type and attributes.

- **delete method**. This method deletes the specified object.
- **clear method**. This method clears the subnodes (elements) of the given object.
- **set_content method**. This method sets the elements of the given object.
- **search method**. This method searches for object information as described in *Search for Object Information* on page 59.

fso Type

An object of type fso contains the state information for a search. Searches can contain multiple pages worth of data. To retrieve subsequent pages, the same fso object is used to retrieve the next set. An fso object has the following attributes.

Attributes	Data Type	Description
path		Search criteria.
limit		Maximum number of results.
order		Attribute to sort on.
result		ID of the `fso` object to use in subsequent calls.
start		Starting sequence of the first item to return. For example, with a limit of 3, a start of 5 would return items 5, 7, and 7 in the search results page.

`locale` Object

The `locale` object is type `FluidObject` and has the following attributes. This object describes where the device is located.

Attributes	Data Type	Description
longitude	String	Longitude of the device location.
latitude	String	Latitude of the device location.

Samples

`router` Client

`deactivate` Request

```
<fcml from="deactivate@portal" to="router@cp7" >
   <deactivate auth="55a0e7adc54cc5c46993238f7ee69df4"/>
</fcml>
```

`is_deleted` Response

```
<fcml from="router@cp7">
   <cp7.router.deactivate.is_deleted type="agent" idle="0"
                                     connection="direct"/>
</fcml>
```

`restart` Method Request

```
<fcml from="snapshot@portal" to="router@cp7">
   <restart a="76913010" b="39912280"/>
</fcml>
```

`reset` Method Request

```
<fcml from="snapshot@portal" to="router@cp7">
   <reset a="08914260" b="28912070"/>
</fcml>
```

`resetting` Response

```
<fcml from="router@cp7" to="router@portal">
   <portal.router.cp7.resetting />
</fcml>
```

`upload` Method Request

```
<fcml from="snapshot@portal" to="router@cp7" _snapshotID="1123765068">
   <_messages.upload
         url "http://ecoportal.netgear.com/diagnostics/LogServlet?cp=cp7"/>
</fcml>
```

`get` Method Request

```
<fcml from="snapshot@portal" to="router@cp7">
   <get/>
</fcml>
```

`is_all` Method Response

```
<fcml from="router@cp7" to="snapshot@portal">
      <cp7.router.sessionkey.is_all type="agent" queued="0" messages="195"
               idle="10038" max_queued="1" connection="direct"/>
      <cp7.router.router.is_all type="router" queued="0" messages="226"
               idle="10043" max_queued="3" connection="direct"/>
      <cp7.router.frs.is_all type="agent" queued="0" messages="221"
               idle="10042" max_queued="1" connection="direct"/>
      <cp7.router.netrouter.is_all type="agent" queued="0" messages="213"
               idle="10041" max_queued="1" connection="direct"/>
      <cp7.router.oma.is_all type="agent" queued="0" messages="200"
               idle="10042" max_queued="1" connection="direct"/>
      <cp7.router.ndw.is_all type="driver" queued="0" messages="219"
               idle="10042" max_queued="1" connection="direct"/>
      <cp7.router.pgw.is_all type="gateway" queued="0" messages="210"
               idle="10039" max_queued="1" connection="direct"/>
      <cp7.router.devices.is_all type="agent" queued="0" messages="218"
               idle= "10042" max_queued="1" connection="direct"/>
      <cp7.router.serialnumber.is_all type="agent" queued="0" messages="199"
               idle="10038" max_queued="1" connection="direct"/>
      <cp7.router.upnp.is_all type="agent" queued="0" messages="179"
               idle="10038" max_queued="1" connection="direct"/>
```

```
   <cp7.router.trig.is_all type="agent" queued="0" messages="223"
            idle="10042" max_queued="1" connection="direct"/>
   <cp7.router.vns.is_all type="store" queued="0" messages="226"
            idle="10043" max_queued="3" connection="direct"/>
   <cp7.router.tweener.is_all type="agent" queued="0" messages="193"
            idle="10038" max_queued="1" connection="direct"/>
   <cp7.router.rmtacs.is_all type="agent" queued="0" messages="199"
            idle= "10038" max_queued="1" connection="direct"/>
   <cp7.router.tss.is_all type="agent" queued="0" messages="53"
            idle="10042" max_queued="1" connection="direct"/>
   <cp7.router.ndc.is_all type="driver" queued="0" messages="214"
            idle="10041" max_queued="5" connection="direct"/>
   <cp7.router.jrs.is_all type="agent" queued="0" messages="219"
            idle="10042" max_queued="1" connection="direct"/>
</fcml>
```

`_ifconfig` Object `get` Request:

```
<fcml from="snapshot@portal" to="router@cp7">
   <_ifconfig.get/>
</fcml>
```

`_ifconfig` Object `is_all` Response

See *Elements* on page 42 for information about the CDATA element.

```
<fcml from="router@cp7" to="snapshot@portal">
   <cp7.router._ifconfig.is_all type="system"
      friendly_name="Ethernet configuration">
   <![CDATA[ /* ifconfig output */ ]]> </cp7.router._ifconfig.is_all>
</fcml>
```

`snapshot` Method Request:

```
<fcml from="snapshot@portal" to="router@cp7">
   <snapshot/>
</fcml>
```

`is_all` Method Response

```
<fcml from="router@cp7" to="snapshot@portal">
   <cp7.router._status.is_all faults="0" rx_errors="" rx_messages=""
         tx_errors="" tx_messages="" uptime="10260" type="system"
         time="1315604895343" friendly_name="Status"/>
   <cp7.router._about.is_all package="fluid-core" version="3.2.27"
         legal="Copyright 2007 4HomeMedia, Inc." type="system"
         description="NETGEAR Control Point" friendly_name="About"
         build_date="20110901 1615"/>
```

```
   <cp7.router._installed_packages.is_all kernel="2.6.21.5" core="3.2.27"
         friendly_name="About"/>
   <cp7.router._product.is_all model_name="WNDR3700"
         manufacturer_url="http://www.Netgear.com/"
         model_description="RangeMax Dual Band Wireless-N Gigabit Router"
         manufacturer_name="Netgear" type="system" friendly_name="Product"
         model_number="CP2000" mac="E091F5D75B7F"/>
   <cp7.router._ps.is_all type="system" friendly_name="Running Processes">
         <![CDATA[  /* followed by snapshot data */ ]]>
   </cp7.router._resolv_conf.is_all>
</fcml>
```

`_config` Object `get` Request

```
<fcml from="snapshot@portal" to="router@cp7">
   <_config.get/>
</fcml>
```

`_config` Object `is_all` Response:

```
<fcml from="router@cp7" to="snapshot@portal">
   <cp7.router._config.is_all http_port="80" type="system" domain="cp7"
         https_enabled="true" https_port="443" />
</fcml>
```

`netrouter` Client

`snapshot` Method Request

```
<fcml from="snapshot@portal" to="netrouter@cp7" _snapshotID="884085758">
   <snapshot/>
</fcml>
```

`is_all` Method Response

```
<fcml from="netrouter@cp7" to="snapshot@portal" _snapshotID="884085758">
   <cp7.netrouter._wlan_setting.is_all enable="0" channel="36"
         wireless_mode="300Mbps" basic_encryption_modes="None"
         wpa_encryption_modes="None" status="Disabled" wlan_mac="E091F5D75B7E"
         region="US" wep_auth_type="Automatic" type="system" ssid="NETGEAR-5G"
         ssid_broadcast="0" friendly_name="WLAN Settings"/>
   <cp7.netrouter.WANIPConnection.GetInfo NewExternalIPAddress="10.1.13.44"
         NewDefaultGateway="10.1.13.13" NewEnable="1"
         NewMACAddress="E091F5D75B7F" NewDNSServers="10.1.1.7 10.1.1.6"
         NewAddressingType="Down" NewSubnetMask="255.255.255.0"
         NewMaxMTUSize="1500" NewMACAddressOverride="0"
         NewConnectionType="Down" NewDNSEnabled="1"
```

```
         type="system" friendly_name="WAN Settings"/>
   <cp7.netrouter._lan_setting.is_all type="system" ip_address="192.168.0.1"
      subnet_mask="&gt;255.255.255.0" friendly_name="LAN Settings"/>
   <cp7.netrouter._attached_devices.is_all mac_0="00:24:7E:DE:2E:91"
      type="system" name_0="JASON-THINKPAD-R400" address_0="192.168.0.3"
      devices="1" friendly_name="Attached Devices"/>
   <cp7.netrouter.DeviceInfo.GetInfo Otherhardwareversion="N/A"
         OthersoftwareVersion="1.0.12NA" VPNVersion="N/A"
         FirstUseDate="Tuesday, 12 Jul 2011 17:39:19"
         Hardwareversion="DGND3700" ModelName="DGND3700"
         FirewallVersion="ACOS NAT-Netfilter v3.0.0.4
         (Linux Cone NAT Hot Patch 03/23/2010)" type="system"
         SmartAgentversion="3.0" Description="Netgear Smart Wizard 3.0,
               specification 0.7 version" Firmwareversion="V1.0.0.12"
         SerialNumber="2KB10C7G00058" friendly_name="Device Info"/>
</fcml>
```

upnp *Client*

`get` Request

```
<fcml from="snapshot@portal" to="upnp@cp7">
   <get />
</fcml>
```

`is_all` Response

```
<fcml from="upnp@cp7" to="snapshot@portal">
   <cp7.upnp.dev1.is_all manufacturer="Microsoft Corporation"
      model_name="Windows Media Player" last_update="1315868126"
      device_type="urn:schemas-upnp-org:device:MediaRenderer:1"
      address="10.1.13.40"
      location= "http://10.1.13.40:2869/upnphost/udhisapi.dll?content=
            uuid:c135c400-5294-4c2f-8a77-42dee0448594"
      active="true" discovered="1315867232129" type="upnp"
      uuid="c135c400-5294-4c2f-8a77-42dee0448594" model_number="12"/>
   <cp7.upnp.dev0.is_all manufacturer="NETGEAR"
      model_name="Windows Media Connect compatible (ReadyDLNA)"
      last_update="1315867819"
      device_type="urn:schemas-upnp-org:device:MediaServer:1"
      address="10.1.13.28" location="http://10.1.13.28:8200/rootDesc.xml"
      active="true" discovered="1315866919586"
      presentation_url="http://10.1.13.28:8200/" type="upnp"
      uuid="4d696e69-444c-164e-9d41-0024b2ad5522" model_number="1"/>
```

```
</fcml>
```

tss Client

get Request

```
<fcml from="snapshot@portal" to="tss@cp7">
   <_config.get/>
</fcml>
```

is_all Response

```
<fcml from="tss@cp7" to="snapshot@portal">
   <cp7.tss._config.is_all type="system" time_zone="UTC"/>
</fcml>
```

set Request

```
<fcml from="snapshot@portal" to="tss@cp7">
   <_config.set time_zone="XYZ"/>
</fcml>
```

is_new Response

```
<fcml from="vns@cp7">
   <cp7.vns.locale.is_new created="1315617915618" modified="1315617915619"
         type="locale" creator="tss@cp7" time_zone="XYZ"/>
</fcml>
```

ndc Client

get Request

```
<fcml to="ndc@cp7" from="snapshot@portal">
   <get/>
</fcml>
```

is_all Response

```
<fcml from="ndc@cp7" to="snapshot@portal">
   <cp7.ndc.host_internet_gateway.is_all wan_upstream_rate=""
         wan_rate_configuration="" wan_downstream_rate=""
         type="internet_gateway" ip_address="10.1.13.13"/>
   <cp7.ndc.host_lan_interface.is_all type="lan_interface"
         configuration="dhcp"/>
</fcml>
```

set Request

```
<fcml to="ndc@cp7" from="snapshot@portal">
```

```
   <_config.set type="internet_gateway" ip_address="10.1.13.13"/>
</fcml>
```

`set` Request

```
<fcml to="ndc@cp7" from="snapshot@portal">
   <_config.set type="lan_interface" configuration="dhcp"/>
</fcml>
```

`configure_wifi` Request

```
<fcml to="ndc@cp7" from="snapshot@portal">
   <configure_wifi device="XXX" type="internet_gateway" ssid="YYY"
      channel="3" security="wep" current_key="ZZZ"/>
</fcml>
```

vns Client

`get` Request

```
<fcml from="snapshot@portal" to="vns@cp7" _queryid="0">
   <get select="trigger" where="trigger!=''"/>
</fcml>
```

`is` Response

```
<fcml from="vns@cp7" to="snapshot@portal" _queryid="0">
   <cp7.vns.home_stay.is trigger="markup">
      <trigger id="596538623">
         <time time="2300" days="1234567"/>
      </trigger>
      <script id="home_stay_0">
         <tweener..new type="tween" end="${state}" attribute="vns.home.state"
               tick="1000" steps="${delay}" algorithm="countdown"/>
      </script>
   </cp7.vns.home_stay.is>
   <cp7.vns.home_away.is trigger="markup">
      <trigger id="1126925078">
         <time time="0800" days="23456"/>
      </trigger>
      <script id="home_away_0">
         <tweener..new type="tween" end="${state}" attribute="vns.home.state"
            tick="1000" steps="${delay}" algorithm="countdown"/>
      </script>
   </cp7.vns.home_away.is>
</fcml>
<fcml from="alerts@portal" to="vns@cp7" >
```

```
   <new urgency="none" type="log" text="THIS IS AN ALERT TEST!!!" level="1"/>
</fcml>
```

new Method Request

```
<fcml from="alerts@portal" to="vns@cp7">
   <new urgency="none" type="log" text="THIS IS AN ALERT TEST!!!" level="1"/>
</fcml>
```

is_new Method Response

```
<fcml from="vns@cp7">
   <cp7.vns.o0.is_new level="1" created="1315608566883"
      modified="1315608566883" urgency="none" type="log"
      creator="alerts@portal" text="THIS IS AN ALERT TEST!!!"/>
</fcml>
```

delete Method Request

```
<fcml from="snapshot@portal" to="vns@cp7">
   <home_home.delete/>
</fcml>
```

is_deleted Method Response

```
<fcml from="vns@cp7">
   <cp7.vns.home_home.is_deleted state="home" delay="30" trigger="markup"
      created="1315423503996" modified="1315423503996" type="home_state"
      creator="test">
   <trigger id="-1080653838">
      <time time="1800" days="23456"/>
   </trigger>
   <trigger id="-1042217402">
      <time time="0600" days="1234567"/>
   </trigger>
   <script id="home_home_0">
      <tweener..new type="tween" end="${state}" attribute="vns.home.state"
         tick="1000" steps="${delay}" algorithm="countdown"/>
   </script>
   </cp7.vns.home_home.is_deleted>
</fcml>
```

set_contents Method Request

```
<fcml from="snapshot@portal" to="vns@cp7">
   <home_stay.set_contents>
      <trigger id="596538623">
         <time time="2345" days="12345678"/>
      </trigger>
```

```
   </home_stay.set_contents>
</fcml>
```

`is_all` Method Response

```
<fcml from="vns@cp7">
   <cp7.vns.home_stay.is_all state="stay" delay="30" trigger="markup"
      created="1315423504023" modified="1315612597658"
      type="home_state" creator="test">
      <trigger id="596538623">
         <time time="2345" days="12345678"/>
      </trigger>
   </cp7.vns.home_stay.is_all>
</fcml>
```

`set` Request

```
<fcml from="activation@portal" to="vns@cp7" >
   <locale.set longitude="-123" latitude="38.5" type="locale"/>
</fcml>
```

`is` Response

```
<fcml from="vns@cp7">
   <cp7.vns.locale.is modified="1315618559448" type="locale"
      latitude="38.5" longitude="-123"/>
</fcml>
```

`get` Request

```
<fcml from="snapshot@portal" to="vns@cp7">
   <get/>
</fcml>
```

`is_all` Response

```
<fcml from="vns@cp7" to="snapshot@portal">
   <cp7.vns.ipc_driver_panasonic_bbhcm581a.is_all
      upnp_model_number="BB-HCM581A,BB-HCM527A"
      mjpeg_stream_user_agent="User-Agent: Mozilla/5.0 (Windows; U; Windows NT
         5.1; en-US; rv:1.8.1.7) Gecko/20070914 Firefox/2.0.0.7"
      focusauto="Direction=FocusAuto" modified="1315423504745"
      name="Panasonic BB-HCM581A" overridesize="Resolution="
      setpreseturl="/nphPresetName" tiltup="Direction=TiltUp"
      mjpeg_stream_path="/nphMotionJpeg?Resolution=192x144"
      image="/SnapshotJPEG?" zoomout="Direction=ZoomWide"
      panright="Direction=PanRight" tiltdown="Direction=TiltDown"
      setpresetsuccess="Success!" panleft="Direction=PanLeft"
      setpresetreqauth="true" created="1315423504744"
```

```
   creator="fakeclient" setpresettext="PresetOperation=Save&amp;
   PresetName=CP1000&amp;Type=Preset&amp;RPeriod=0&amp;
   PreSave=+++Save+++&amp;Data="
   implements="stream,pan,tilt,zoom,focus,preset"
   recallpreset="/nphControlCamera?Direction=Preset&amp;
   PresetOperation=Move&amp;RPeriod=0&amp;Data="
   controlurl="/nphControlCamera?" focusout="Direction=FocusFar&amp;
   Dist=1" zoomin="Direction=ZoomTele" type="ipc_driver"
   focusin="Direction=FocusNear&amp;Dist=1" >
   <size id="ipc_driver_panasonic_bbhcm581a_0" command="160x120"
      width="160" height="120"/>
   <size id="ipc_driver_panasonic_bbhcm581a_1" command="320x240"
      width="320" height="240"/>
   <size id="ipc_driver_panasonic_bbhcm581a_2" command="640x480"
      width="640" height="480"/>
</cp7.vns.ipc_driver_panasonic_bbhcm581a.is_all>
<cp7.vns.ipc_driver_dlink_dcs900.is_all controlurl=""
   upnp_model_name="DCS-900,DCS-G900" created="1315423504727"
   modified="1315423504727" type="ipc_driver" name="D-Link DCS-900"
   image="/IMAGE.JPG" creator="fakeclient" overridesize="">
   <size id="ipc_driver_dlink_dcs900_0" command="" width="320"
      height="240"/>
</cp7.vns.ipc_driver_dlink_dcs900.is_all>
<cp7.vns.home_stay.is_all state="stay" delay="30" trigger="markup"
   created="1315423504023" modified="1315423504024" type="home_state"
   creator="test">
   <trigger id="596538623">
      <time time="2300" days="1234567"/>
   </trigger>
   <script id="home_stay_0">
      <tweener..new type="tween" end="${state}" attribute="vns.home.state"
         tick="1000" steps="${delay}" algorithm="countdown"/>
   </script>
</cp7.vns.home_stay.is_all>
<cp7.vns.ipc_driver_sercom_rc4020.is_all upnp_model_number="RC4020"
   rtsp_stream_path="/img/media.sav" created="1315423504867"
   http_auth_default_user="administrator" modified="1315423504867"
   name="Sercom RC4020" versionpre="Firmware Version: V"
   creator="fakeclient" overridesize="size="
   check_auth_url="/adm/sysinfo.cgi"
   implements="stream,record,version,reboot"
   rtsp_user_agent="Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path
```

```
   2)" reboot_url="/adm/reboot.cgi"
versionurl="/adm/sysinfo.cgi" asf_stream_path="/img/video.asf"
http_auth_default_password="" sensors="motion,sound" type="ipc_driver"
mjpeg_stream_path="/img/video.mjpeg" image="/img/snapshot.cgi?">
<brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
   brightness_1="1" brightness_2="2" brightness_3="3"
   brightness_4="4" brightness_5="5" brightness_6="6"
   brightness_7="7" id="ipc_driver_sercom_rc4020_0" />
<sharpness sharpness_url="/adm/file.cgi?todo=save&amp;h_sharpness="
   sharpness_1="1" sharpness_2="2" sharpness_3="3" sharpness_4="4"
   sharpness_5="5" sharpness_6="6" sharpness_7="7"
   id="ipc_driver_sercom_rc4020_1"/>
<flip flip_true="1" flip_false="0" id="ipc_driver_sercom_rc4020_2"
   flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
<mirror id="ipc_driver_sercom_rc4020_3" mirror_false="0"
   mirror_true="1"
   mirror_url="/adm/set_group.cgi?group=VIDEO&amp;mirror="/>
<overlay_text overlay_text_url="/adm/set_group.cgi?group=
   VIDEO&amp;text_overlay=1&amp;text="
   id="ipc_driver_sercom_rc4020_4"/>
<overlay_timestamp id="ipc_driver_sercom_rc4020_5"
   overlay_timestamp_false="0" overlay_timestamp_true="1"
   overlay_timestamp_url="/adm/set_group.cgi?group=
      VIDEO&amp;time_stamp="/>
<white_balance white_balance_auto="0" id="ipc_driver_sercom_rc4020_6"
   white_balance_daylight="4" white_balance_tungsten="1"
   white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
   white_balance_fluorescent="2"/>
<low_light low_light_true="1" id="ipc_driver_sercom_rc4020_7"
   low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
   low_light_false="0"/>
<audio_encoder audio_enable_false="0" audio_enable_true="1"
   id="ipc_driver_sercom_rc4020_8" audio_check_pre="audio_in="
   audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
   audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
<video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
   bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
   supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
   supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
   quality_level_url="quality_type=1&amp;quality_level="
   supported_resolutions="160x120,320x240,640x480"
   id="ipc_driver_sercom_rc4020_9" supported_framerate="10,20,30"
```

```
      resolution_url="resolution="/>
   <video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
      frame_rate_url="frame_rate=" encoding="mjpeg"
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_sercom_rc4020_10" resolution_url="resolution="
      resolution="" supported_framerate="10,20,30"/>
   <size id="ipc_driver_sercom_rc4020_11" command="1" mjpeg="1"
      width="160" mpeg4="1" height="120"/>
   <size id="ipc_driver_sercom_rc4020_12" command="2" mjpeg="2"
      width="320" mpeg4="2" height="240"/>
   <size id="ipc_driver_sercom_rc4020_13" command="3" mjpeg="3"
      width="640" mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_sercom_rc4020.is_all>
<cp7.vns.vns_init.is_all created="1315423503954" modified="1315423504608"
   type="vns_init" homestate_version="7" creator="test" ipc_version="43" />
<cp7.vns.ipc_driver_sercom_rc8021.is_all upnp_model_number="RC8021"
   wifisecwpaaes="2" wifisecwpa2tkip="2"
   wifikeyurl="/adm/set_group.cgi?group=WIRELESS&amp;wpa_ascii="
   wifisecwep="1" http_auth_default_user="administrator"
   modified="1315423505361" versionpre="Firmware Version: V"
   name="Sercom RC8021" enableurl="/util/query.cgi" overridesize="size="
   wifiiwepkeyindexurl="/adm/set_group.cgi?group=WIRELESS&amp;wep_index="
   reboot_url="/adm/reboot.cgi" versionurl="/adm/sysinfo.cgi"
   http_auth_default_password="" mjpeg_stream_path="/img/video.mjpeg"
   image="/img/snapshot.cgi?" wifissidurl="/adm/set_group.cgi?group=
      WIRELESS&amp;wlan_essid=" wifichannelurl="/adm/set_group.cgi?group=
      WIRELESS&amp;wlan_channel=" rtsp_stream_path="/img/media.sav"
   wifisecopen="0" created="1315423505361" wifisecwpa2aes="2"
   wifisecwpatkip="2" creator="fakeclient"
   check_auth_url="/adm/sysinfo.cgi" enabletext="privacy_button=on"
   implements="stream,enable,version,record,wifi,reboot,install"
   rtsp_user_agent="Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path
      2)" sensors="motion,sound" asf_stream_path="/img/video.asf"
   type="ipc_driver" wifisecurityurl="/adm/set_group.cgi?group=
      WIRELESS&amp;wlan_security=">
   mirror_url="/adm/set_group.cgi?group=VIDEO&amp;mirror="/>
   <overlay_text overlay_text_url="/adm/set_group.cgi?group=
      VIDEO&amp;text_overlay=1&amp;text=" id="ipc_driver_sercom_rc8021_4"/>
   <overlay_timestamp id="ipc_driver_sercom_rc8021_5"
      overlay_timestamp_false="0" overlay_timestamp_true="1"
      overlay_timestamp_url="/adm/set_group.cgi?group=VIDEO&amp;time_stamp=
      "/>
```

```
    <white_balance white_balance_auto="0" id="ipc_driver_sercom_rc8021_6"
       white_balance_daylight="4" white_balance_tungsten="1"
       white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
       white_balance_fluorescent="2"/>
    <low_light low_light_true="1" id="ipc_driver_sercom_rc8021_7"
       low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
       low_light_false="0"/>
    <audio_encoder audio_enable_false="0" audio_enable_true="1"
       id="ipc_driver_sercom_rc8021_8" audio_check_pre="audio_in="
       audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
       audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
    <video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
       bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
       supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
       supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
       quality_level_url="quality_type=1&amp;quality_level="
       supported_resolutions="160x120,320x240,640x480"
       id="ipc_driver_sercom_rc8021_9" supported_framerate="10,20,30"
       resolution_url="resolution="/>
    <video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
       frame_rate_url="frame_rate=" encoding="mjpeg"
       supported_resolutions="160x120,320x240,640x480"
       id="ipc_driver_sercom_rc8021_10" resolution_url="resolution="
       resolution="" supported_framerate="10,20,30"/>
    <size id="ipc_driver_sercom_rc8021_11" command="1" mjpeg="1"
       width="160" mpeg4="1" height="120"/>
    <size id="ipc_driver_sercom_rc8021_12" command="2" mjpeg="2"
       width="320" mpeg4="2" height="240"/>
    <size id="ipc_driver_sercom_rc8021_13" command="3" mjpeg="3"
       width="640" mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_sercom_rc8021.is_all>
<cp7.vns.ipc_driver_sercom_rc8020.is_all upnp_model_number="RC8020"
   rtsp_stream_path="/img/media.sav" created="1315423505223"
   http_auth_default_user="administrator" modified="1315423505223"
   name="Sercom RC8020" versionpre="Firmware Version: V"
   creator="fakeclient" overridesize="size="
   check_auth_url="/adm/sysinfo.cgi"
   implements="stream,version,record,reboot,install"
   rtsp_user_agent="Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path
      2)" reboot_url="/adm/reboot.cgi" versionurl="/adm/sysinfo.cgi"
   asf_stream_path="/img/video.asf" http_auth_default_password=""
   sensors="motion,sound" type="ipc_driver"
```

```
mjpeg_stream_path="/img/video.mjpeg" image="/img/snapshot.cgi?">
<brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
   brightness_1="1" brightness_2="2" brightness_3="3" brightness_4="4"
   brightness_5="5" brightness_6="6" brightness_7="7"
   id="ipc_driver_sercom_rc8020_0"/>
<sharpness sharpness_url="/adm/file.cgi?todo=save&amp;h_sharpness="
   sharpness_1="1" sharpness_2="2" sharpness_3="3" sharpness_4="4"
   sharpness_5="5" sharpness_6="6" sharpness_7="7"
   id="ipc_driver_sercom_rc8020_1"/>
<flip flip_true="1" flip_false="0" id="ipc_driver_sercom_rc8020_2"
   flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
<mirror id="ipc_driver_sercom_rc8020_3" mirror_false="0" mirror_true="1"
   mirror_url="/adm/set_group.cgi?group=VIDEO&amp;mirror="/>
<overlay_text overlay_text_url="/adm/set_group.cgi?group=
   VIDEO&amp;text_overlay=1&amp;text=" id="ipc_driver_sercom_rc8020_4"/>
<overlay_timestamp id="ipc_driver_sercom_rc8020_5"
   overlay_timestamp_false="0" overlay_timestamp_true="1"
   overlay_timestamp_url="/adm/set_group.cgi?group=VIDEO&amp;time_stamp=
      "/>
<white_balance white_balance_auto="0" id="ipc_driver_sercom_rc8020_6"
   white_balance_daylight="4" white_balance_tungsten="1"
   white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
   white_balance_fluorescent="2"/>
<low_light low_light_true="1" id="ipc_driver_sercom_rc8020_7"
   low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
   low_light_false="0"/>
<audio_encoder audio_enable_false="0" audio_enable_true="1"
   id="ipc_driver_sercom_rc8020_8" audio_check_pre="audio_in="
   audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in=" a
   udio_check_url="/adm/get_group.cgi?group=AUDIO"/>
<video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
   bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
   supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
   supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
   quality_level_url="quality_type=1&amp;quality_level="
   supported_resolutions="160x120,320x240,640x480"
   id="ipc_driver_sercom_rc8020_9" supported_framerate="10,20,30"
   resolution_url="resolution="/>
<video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
   frame_rate_url="frame_rate=" encoding="mjpeg"
   supported_resolutions="160x120,320x240,640x480"
   id="ipc_driver_sercom_rc8020_10" resolution_url="resolution="
```

```
      resolution="" supported_framerate="10,20,30"/>
   <size id="ipc_driver_sercom_rc8020_11" command="1" mjpeg="1" width="160"
      mpeg4="1" height="120"/>
   <size id="ipc_driver_sercom_rc8020_12" command="2" mjpeg="2"
      width="320" mpeg4="2" height="240"/>
   <size id="ipc_driver_sercom_rc8020_13" command="3" mjpeg="3" width="640"
      mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_sercom_rc8020.is_all>
<cp7.vns.ipc_driver_7521ff.is_all upnp_model_number="7521FF"
   wifisecwpaaes="2" wifisecwpa2tkip="2"
   wifikeyurl="/adm/set_group.cgi?group=WIRELESS&amp;wpa_ascii="
   wifisecwep="1" http_auth_default_user="administrator"
   modified="1315423505288" versionpre="Firmware Version: W"
   name="Wireless IP Camera" enableurl="/util/query.cgi"
   overridesize="size=" wifiiwepkeyindexurl="/adm/set_group.cgi?group=
      WIRELESS&amp;wep_index=" reboot_url="/adm/reboot.cgi"
   versionurl="/adm/sysinfo.cgi" http_auth_default_password=""
   mjpeg_stream_path="/img/video.mjpeg" image="/img/snapshot.cgi?"
   wifissidurl="/adm/set_group.cgi?group=WIRELESS&amp;wlan_essid="
   wifichannelurl="/adm/set_group.cgi?group=WIRELESS&amp;wlan_channel="
   rtsp_stream_path="/img/media.sav" wifisecopen="0"
   created="1315423505288" wifisecwpa2aes="2" wifisecwpatkip="2"
   creator="fakeclient" check_auth_url="/adm/sysinfo.cgi"
   enabletext="privacy_button=on"
   implements="stream,enable,version,record,wifi,reboot,install"
   rtsp_user_agent="Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path
      2)" sensors="motion,sound" asf_stream_path="/img/video.asf"
   type="ipc_driver" wifisecurityurl="/adm/set_group.cgi?group=
      WIRELESS&amp;wlan_security=">
   <brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
      brightness_1="1" brightness_2="2" brightness_3="3" brightness_4="4"
      brightness_5="5" brightness_6="6" brightness_7="7"
      id="ipc_driver_7521ff_0"/>
   <sharpness sharpness_url="/adm/file.cgi?todo=save&amp;h_sharpness="
      sharpness_1="1" sharpness_2="2" sharpness_3="3" sharpness_4="4"
      sharpness_5="5" sharpness_6="6" sharpness_7="7"
      id="ipc_driver_7521ff_1"/>
   <flip flip_true="1" flip_false="0" id="ipc_driver_7521ff_2"
      flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
   <mirror id="ipc_driver_7521ff_3" mirror_false="0" mirror_true="1"
      mirror_url="/adm/set_group.cgi?group=VIDEO&amp;mirror="/>
   <overlay_text overlay_text_url="/adm/set_group.cgi?group=
```

```
      VIDEO&amp;text_overlay=1&amp;text=" id="ipc_driver_7521ff_4"/>
   <overlay_timestamp id="ipc_driver_7521ff_5" overlay_timestamp_false="0"
      overlay_timestamp_true="1" overlay_timestamp_url=
         "/adm/set_group.cgi?group=VIDEO&amp;time_stamp="/>
   <white_balance white_balance_auto="0" id="ipc_driver_7521ff_6"
      white_balance_daylight="4" white_balance_tungsten="1"
      white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
      white_balance_fluorescent="2"/>
   <low_light low_light_true="1" id="ipc_driver_7521ff_7"
      low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
      low_light_false="0"/>
   <audio_encoder audio_enable_false="0" audio_enable_true="1"
      id="ipc_driver_7521ff_8" audio_check_pre="audio_in="
      audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
      audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
   <video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
      bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
      supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
      supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
      quality_level_url="quality_type=1&amp;
      quality_level=" supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_7521ff_9" supported_framerate="10,20,30"
      resolution_url="resolution="/>
   <video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
      frame_rate_url="frame_rate=" encoding="mjpeg"
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_7521ff_10" resolution_url="resolution=" resolution=""
      supported_framerate="10,20,30"/>
   <size id="ipc_driver_7521ff_11" command="1" mjpeg="1" width="160"
      mpeg4="1" height="120"/>
   <size id="ipc_driver_7521ff_12" command="2" mjpeg="2" width="320"
      mpeg4="2" height="240"/>
   <size id="ipc_driver_7521ff_13" command="3" mjpeg="3" width="640"
      mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_7521ff.is_all>
<cp7.vns.ipc_driver_sercom_rc8030.is_all upnp_model_number="RC8030"
   http_auth_default_user="administrator" modified="1315423505432"
   name="Sercom RC8030" versionpre="Firmware Version: V"
   overridesize="size=" setpreseturl="/pt/ptctrl.cgi?preset=set,"
   reboot_url="/adm/reboot.cgi" versionurl="/adm/sysinfo.cgi"
   tiltup="mv=U,5,5" http_auth_default_password=""
   mjpeg_stream_path="/img/video.mjpeg" image="/img/snapshot.cgi?"
```

```
tiltdown="mv=D,5,5" panright="mv=R,5,5" panleft="mv=L,5,5"
rtsp_stream_path="/img/media.sav" setpresetreqauth="true"
created="1315423505432" creator="fakeclient"
check_auth_url="/adm/sysinfo.cgi"
implements="pan,tilt,stream,preset,version,record,reboot,wifi,install"
recallpreset="/pt/ptctrl.cgi?preset=move,"
rtsp_user_agent="Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service
   Path 2)" controlurl="/pt/ptctrl.cgi?" sensors="motion,sound"
asf_stream_path="/img/video.asf" type="ipc_driver">
<brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
   brightness_1="1" brightness_2="2" brightness_3="3" brightness_4="4"
   brightness_5="5" brightness_6="6" brightness_7="7"
   id="ipc_driver_sercom_rc8030_0"/>
<sharpness sharpness_url="/adm/file.cgi?todo=save&amp;
   h_sharpness=" sharpness_1="1" sharpness_2="2" sharpness_3="3"
   sharpness_4="4" sharpness_5="5" sharpness_6="6" sharpness_7="7"
   id="ipc_driver_sercom_rc8030_1"/>
<flip flip_true="1" flip_false="0" id="ipc_driver_sercom_rc8030_2"
   flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
<mirror id="ipc_driver_sercom_rc8030_3" mirror_false="0" mirror_true="1"
   mirror_url="/adm/set_group.cgi?group=VIDEO&amp;mirror="/>
<overlay_text overlay_text_url="/adm/set_group.cgi?group=
   VIDEO&amp;text_overlay=1&amp;text=" id="ipc_driver_sercom_rc8030_4"/>
<overlay_timestamp id="ipc_driver_sercom_rc8030_5"
   overlay_timestamp_false="0" overlay_timestamp_true="1"
   overlay_timestamp_url="/adm/set_group.cgi?group=VIDEO&amp;time_stamp=
      "/>
<white_balance white_balance_auto="0" id="ipc_driver_sercom_rc8030_6"
   white_balance_daylight="4" white_balance_tungsten="1"
   white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
   white_balance_fluorescent="2"/>
<low_light low_light_true="1" id="ipc_driver_sercom_rc8030_7"
   low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
   low_light_false="0"/>
<audio_encoder audio_enable_false="0" audio_enable_true="1"
   id="ipc_driver_sercom_rc8030_8" audio_check_pre="audio_in="
   audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
   audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
<video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
   bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
   supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
   supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
```

```
      quality_level_url="quality_type=1&amp;quality_level="
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_sercom_rc8030_9" supported_framerate="10,20,30"
      resolution_url="resolution="/>
   <video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
      frame_rate_url="frame_rate=" encoding="mjpeg"
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_sercom_rc8030_10" resolution_url="resolution="
      resolution="" supported_framerate="10,20,30"/>
   <size id="ipc_driver_sercom_rc8030_11" command="1" mjpeg="1"
         width="160" mpeg4="1" height="120"/>
   <size id="ipc_driver_sercom_rc8030_12" command="2" mjpeg="2"
      width="320" mpeg4="2" height="240"/>
   <size id="ipc_driver_sercom_rc8030_13" command="3" mjpeg="3" width="640"
      mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_sercom_rc8030.is_all>
<cp7.vns.ipc_driver_axis_207.is_all implements="stream"
   controlurl="/axis-cgi/com/ptz.cgi?" upnp_model_name="AXIS 207"
   created="1315423504630" modified="1315423504630" type="ipc_driver"
   mjpeg_stream_path="/mjpg/video.mjpg" name="Axis 207"
   image="/axis-cgi/jpg/image.cgi?" creator="fakeclient" o
   verridesize="resolution=">
   <size id="ipc_driver_axis_207_0" command="160x120" width="160"
      height="120"/>
   <size id="ipc_driver_axis_207_1" command="320x240" width="320"
      height="240"/>
   <size id="ipc_driver_axis_207_2" command="640x480" width="640"
      height="480"/>
</cp7.vns.ipc_driver_axis_207.is_all>
<cp7.vns.ipc_driver_sercom_rc8061.is_all upnp_model_number="RC8061"
   http_auth_default_user="administrator" modified="1315423505504"
   name="Sercom RC8061" versionpre="Firmware Version: V"
   overridesize="size=" setpreseturl="/pt/ptctrl.cgi?preset=set,"
   reboot_url="/adm/reboot.cgi" versionurl="/adm/sysinfo.cgi"
   tiltup="mv=U,5,5" http_auth_default_password=""
   mjpeg_stream_path="/img/video.mjpeg" image="/img/snapshot.cgi?"
   tiltdown="mv=D,5,5" panright="mv=R,5,5" panleft="mv=L,5,5"
   rtsp_stream_path="/img/media.sav" setpresetreqauth="true"
   created="1315423505504" creator="fakeclient"
   check_auth_url="/adm/sysinfo.cgi"
   implements="pan,tilt,stream,preset,version,record,reboot,wifi,install"
   recallpreset="/pt/ptctrl.cgi?preset=move," rtsp_user_agent=
```

```
   "Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path 2)"
controlurl="/pt/ptctrl.cgi?" sensors="motion,pir,led" asf_stream_path=
   "/img/video.asf" type="ipc_driver">
<brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
   brightness_1="1" brightness_2="2" brightness_3="3" brightness_4="4"
   brightness_5="5" brightness_6="6" brightness_7="7"
   id="ipc_driver_sercom_rc8061_0"/>
<sharpness sharpness_url="/adm/file.cgi?todo=save&amp;h_sharpness="
   sharpness_1="1" sharpness_2="2" sharpness_3="3" sharpness_4="4"
   sharpness_5="5" sharpness_6="6" sharpness_7="7"
   id="ipc_driver_sercom_rc8061_1"/>
<flip flip_true="1" flip_false="0" id="ipc_driver_sercom_rc8061_2"
   flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
<mirror id="ipc_driver_sercom_rc8061_3" mirror_false="0" mirror_true="1"
   mirror_url="/adm/set_group.cgi?group=VIDEO&amp;mirror="/>
<overlay_text overlay_text_url="/adm/set_group.cgi?group=
   VIDEO&amp;text_overlay=1&amp;text=" id="ipc_driver_sercom_rc8061_4"/>
<overlay_timestamp id="ipc_driver_sercom_rc8061_5"
   overlay_timestamp_false="0" overlay_timestamp_true
   overlay_timestamp_url=
   "/adm/set_group.cgi?group=VIDEO&amp;time_stamp="/>
<white_balance white_balance_auto="0" id="ipc_driver_sercom_rc8061_6"
   white_balance_daylight="4" white_balance_tungsten="1"
   white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
   white_balance_fluorescent="2"/>
<low_light low_light_true="1" id="ipc_driver_sercom_rc8061_7"
   low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
   low_light_false="0"/>
<audio_encoder audio_enable_false="0" audio_enable_true="1"
   id="ipc_driver_sercom_rc8061_8" audio_check_pre="audio_in="
   audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
   audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
<video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
   bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
   supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
   supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
   quality_level_url="quality_type=1&amp;quality_level="
   supported_resolutions="160x120,320x240,640x480"
   id="ipc_driver_sercom_rc8061_9" supported_framerate="10,20,30"
   resolution_url="resolution="/>
<video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
   frame_rate_url="frame_rate=" encoding="mjpeg"
```

```
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_sercom_rc8061_10" resolution_url="resolution="
      resolution="" supported_framerate="10,20,30"/>
   <size id="ipc_driver_sercom_rc8061_11" command="1" mjpeg="1" width="160"
      mpeg4="1" height="120"/>
   <size id="ipc_driver_sercom_rc8061_12" command="2" mjpeg="2" width="320"
      mpeg4="2" height="240"/>
   <size id="ipc_driver_sercom_rc8061_13" command="3" mjpeg="3" width="640"
      mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_sercom_rc8061.is_all>
<cp7.vns.home_away.is_all state="away" delay="30" trigger="markup"
   created="1315423504057" modified="1315423504057" type="home_state"
   creator="test">
   <trigger id="1126925078">
      <time time="0800" days="23456"/>
   </trigger>
   <script id="home_away_0">
      <tweener..new type="tween" end="${state}" attribute="vns.home.state"
         tick="1000" steps="${delay}" algorithm="countdown"/>
   </script>
</cp7.vns.home_away.is_all>
<cp7.vns.ipc_driver_panasonic_blc10.is_all u
   pnp_model_number="BL-C10,BL-C30A,BL-C111A,BL-C131A"
   setpresetsuccess="Success!" panleft="Direction=PanLeft"
   setpresetreqauth="true" mjpeg_stream_user_agent="User-Agent: Mozilla/5.0
      (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.7) Gecko/20070914
      Firefox/2.0.0.7" created="1315423504816" modified="1315423504816"
      name="Panasonic BL-C10" creator="fakeclient"
   overridesize="Resolution=" setpresettext="
   PresetOperation=Save&amp;PresetName=CP1000&amp;Type=Preset&amp;
   RPeriod=0&amp;PreSave=+++Save+++&amp;Data="
   setpreseturl="/nphPresetName" implements="pan,tilt,preset"
   recallpreset="/nphControlCamera?Direction=Preset&amp;
   PresetOperation=Move&amp;RPeriod=0&amp;Data="
   controlurl="/nphControlCamera?" tiltup="Direction=TiltUp"
   type="ipc_driver" mjpeg_stream_path="/nphMotionJpeg?Resolution=192x144"
   image="/SnapshotJPEG?" panright="Direction=PanRight"
   tiltdown="Direction=TiltDown">
   <size id="ipc_driver_panasonic_blc10_0" command="160x120" width="160"
      height="120"/>
   <size id="ipc_driver_panasonic_blc10_1" command="320x240" width="320"
      height="240"/>
```

```
    <size id="ipc_driver_panasonic_blc10_2" command="640x480" width="640"
       height="480"/>
</cp7.vns.ipc_driver_panasonic_blc10.is_all>
<cp7.vns.ipc_driver_linksys_wvc210.is_all upnp_model_number="WVC210"
    http_auth_default_user="admin" modified="1315423505158"
    name="Linksys WVC210" versionpre="Firmware Version: V"
    overridesize="size=" setpreseturl="/pt/ptctrl.cgi?preset=set,"
    reboot_url="/adm/reboot.cgi" versionurl="/adm/sysinfo.cgi"
    tiltup="mv=U,5,5" http_auth_default_password="admin"
    mjpeg_stream_path="/img/video.mjpeg" image="/img/snapshot.cgi?"
    tiltdown="mv=D,5,5" panright="mv=R,5,5" panleft="mv=L,5,5"
    rtsp_stream_path="/img/media.sav" setpresetreqauth="true"
    created="1315423505158" creator="fakeclient"
    check_auth_url="/adm/sysinfo.cgi"
    implements="pan,tilt,stream,preset,version,record,reboot,wifi,install"
    recallpreset="/pt/ptctrl.cgi?preset=move,"
    rtsp_user_agent="Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path
       2)" controlurl="/pt/ptctrl.cgi?" sensors="motion,sound"
       asf_stream_path="/img/video.asf" type="ipc_driver">
    <brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
       brightness_1="1" brightness_2="2" brightness_3="3" brightness_4="4"
       brightness_5="5" brightness_6="6" brightness_7="7"
       id="ipc_driver_linksys_wvc210_0"/>
    <sharpness sharpness_url="/adm/file.cgi?todo=save&amp;h_sharpness="
       sharpness_1="1" sharpness_2="2" sharpness_3="3" sharpness_4="4"
       sharpness_5="5" sharpness_6="6" sharpness_7="7"
       id="ipc_driver_linksys_wvc210_1"/>
    <flip flip_true="1" flip_false="0" id="ipc_driver_linksys_wvc210_2"
       flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
    <mirror id="ipc_driver_linksys_wvc210_3" mirror_false="0"
       mirror_true="1" mirror_url="/adm/set_group.cgi?group=
       VIDEO&amp;mirror="/>
    <overlay_text overlay_text_url="/adm/set_group.cgi?group=
       VIDEO&amp;text_overlay=1&amp;text="
       id="ipc_driver_linksys_wvc210_4"/>
    <overlay_timestamp id="ipc_driver_linksys_wvc210_5"
       overlay_timestamp_false="0" overlay_timestamp_true="1"
       overlay_timestamp_url="/adm/set_group.cgi?group=VIDEO&amp;time_stamp=
          "/>
    <white_balance white_balance_auto="0" id="ipc_driver_linksys_wvc210_6"
       white_balance_daylight="4" white_balance_tungsten="1"
       white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
```

```
        white_balance_fluorescent="2"/>
    <low_light low_light_true="1" id="ipc_driver_linksys_wvc210_7"
        low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
        low_light_false="0"/>
    <audio_encoder audio_enable_false="0" audio_enable_true="1"
        id="ipc_driver_linksys_wvc210_8" audio_check_pre="audio_in="
        audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
        audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
    <video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
        bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
        supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
        supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
        quality_level_url="quality_type=1&amp;quality_level="
        supported_resolutions="160x120,320x240,640x480"
        id="ipc_driver_linksys_wvc210_9" supported_framerate="10,20,30"
        resolution_url="resolution="/>
    <video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
        frame_rate_url="frame_rate=" encoding="mjpeg"
        supported_resolutions="160x120,320x240,640x480"
        id="ipc_driver_linksys_wvc210_10" resolution_url="resolution="
        resolution="" supported_framerate="10,20,30"/>
    <size id="ipc_driver_linksys_wvc210_11" command="1" mjpeg="1"
        width="160" mpeg4="1" height="120"/>
    <size id="ipc_driver_linksys_wvc210_12" command="2" mjpeg="2"
        width="320" mpeg4="2" height="240"/>
    <size id="ipc_driver_linksys_wvc210_13" command="3" mjpeg="3"
        width="640" mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_linksys_wvc210.is_all>
<cp7.vns.ipc_driver_panasonic_blc20.is_all implements="stream"
    upnp_model_number="BL-C20A, BL-C1A" mjpeg_stream_user_agent="User-Agent:
        Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.7)
        Gecko/20070914 Firefox/2.0.0.7" created="1315423504839"
    modified="1315423504839" type="ipc_driver"
    mjpeg_stream_path="/nphMotionJpeg?Resolution=192x144"
    name="Panasonic BL-C20A" image="/SnapshotJPEG?"
    creator="fakeclient" overridesize="Resolution=">
    <size id="ipc_driver_panasonic_blc20_0" command="160x120" width="160"
        height="120"/>
    <size id="ipc_driver_panasonic_blc20_1" command="320x240" width="320"
        height="240"/>
    <size id="ipc_driver_panasonic_blc20_2" command="640x480" width="640"
        height="480"/>
```

```
</cp7.vns.ipc_driver_panasonic_blc20.is_all>
<cp7.vns.ipc_driver_axis_212.is_all upnp_model_name="AXIS 212 PTZ,AXIS 213"
   panleft="move=left" created="1315423504671" http_auth_default_user=
   "root" modified="1315423504671" name="Axis 212/213" creator="fakeclient"
      overridesize="resolution=" implements="pan,tilt,zoom,stream"
      controlurl="/axis-cgi/com/ptz.cgi?camera=1&amp;" zoomin="rzoom=1000"
      tiltup="move=up" http_auth_default_password="pass" mjpeg_stream_path=
      "/mjpg/video.mjpg" type="ipc_driver" zoomout="rzoom=-1000" image=
      "/axis-cgi/jpg/image.cgi?" panright="move=right" tiltdown="move=
      down">
   <size id="ipc_driver_axis_212_0" command="160x120" width="160"
      height="120"/>
   <size id="ipc_driver_axis_212_1" command="176x144" width="176"
      height="144"/>
   <size id="ipc_driver_axis_212_2" command="240x180" width="240"
      height="180"/>
   <size id="ipc_driver_axis_212_3" command="320x240" width="320"
      height="240"/>
   <size id="ipc_driver_axis_212_4" command="480x360" width="480"
      height="360"/>
   <size id="ipc_driver_axis_212_5" command="640x480" width="640"
      height="480"/>
</cp7.vns.ipc_driver_axis_212.is_all>
<cp7.vns.ipc_driver_axis_210.is_all implements="stream"
      controlurl="/axis-cgi/com/ptz.cgi?" upnp_model_name="AXIS 210"
      created="1315423504649" modified="1315423504649" type="ipc_driver"
      mjpeg_stream_path="/mjpg/video.mjpg" name="Axis 210"
      image="/axis-cgi/jpg/image.cgi?" creator="fakeclient"
      overridesize="resolution=">
   <size id="ipc_driver_axis_210_0" command="160x120" width="160"
      height="120"/>
<size id="ipc_driver_axis_210_1" command="320x240" width="320"
      height="240"/>
<size id="ipc_driver_axis_210_2" command="640x480" width="640"
      height="480"/>
</cp7.vns.ipc_driver_axis_210.is_all>
<cp7.vns.o1.is_all level="1" created="1315609054586"
   modified="1315609054586" urgency="none" type="log"
   creator="alerts@portal" text="THIS IS AN ALERT TEST!!!"/>
<cp7.vns.ipc_driver_linksys_wvc54gca.is_all upnp_model_number="WVC54GCA"
   rtsp_stream_path="/img/media.sav" created="1315423505087"
   http_auth_default_user="admin" modified="1315423505088" name="Linksys
```

```
   WVC54GCA" versionpre="Firmware Version: V" creator="fakeclient"
overridesize="size=" check_auth_url="/adm/sysinfo.cgi"
implements="stream,version,record,reboot,install" rtsp_user_agent=
   "Quicktime/7.2 (qtver=7.2;os=windows NT 5.1Service Path 2)"
reboot_url="/adm/reboot.cgi" versionurl="/adm/sysinfo.cgi"
asf_stream_path="/img/video.asf" http_auth_default_password="admin"
sensors="motion,sound" type="ipc_driver" mjpeg_stream_path=
   "/img/video.mjpeg" image="/img/snapshot.cgi?">
<brightness brightness_url="/adm/file.cgi?todo=save&amp;h_exposure="
   brightness_1="1" brightness_2="2" brightness_3="3" brightness_4="4"
   brightness_5="5" brightness_6="6" brightness_7="7"
   id="ipc_driver_linksys_wvc54gca_0"/>
<sharpness sharpness_url="/adm/file.cgi?todo=save&amp;h_sharpness="
   sharpness_1="1" sharpness_2="2" sharpness_3="3" sharpness_4="4"
   sharpness_5="5" sharpness_6="6" sharpness_7="7"
   id="ipc_driver_linksys_wvc54gca_1"/>
<flip flip_true="1" flip_false="0" id="ipc_driver_linksys_wvc54gca_2"
   flip_url="/adm/set_group.cgi?group=VIDEO&amp;flip="/>
   <mirror id="ipc_driver_linksys_wvc54gca_3" mirror_false="0"
   mirror_true="1" mirror_url="/adm/set_group.cgi?group=
      VIDEO&amp;mirror="/>
<overlay_text overlay_text_url="/adm/set_group.cgi?group=
   VIDEO&amp;text_overlay=1&amp;text="
id="ipc_driver_linksys_wvc54gca_4"/>
<overlay_timestamp id="ipc_driver_linksys_wvc54gca_5"
   overlay_timestamp_false="0" overlay_timestamp_true="1"
   overlay_timestamp_url="/adm/set_group.cgi?group=VIDEO&amp;time_stamp=
      "/>
<white_balance white_balance_auto="0" id="ipc_driver_linksys_wvc54gca_6"
   white_balance_daylight="4" white_balance_tungsten="1"
   white_balance_url="/adm/set_group.cgi?group=VIDEO&amp;color="
   white_balance_fluorescent="2"/>
<low_light low_light_true="1" id="ipc_driver_linksys_wvc54gca_7"
   low_light_url="/adm/set_group.cgi?group=VIDEO&amp;night_mode="
   low_light_false="0"/>
<audio_encoder audio_enable_false="0" audio_enable_true="1"
   id="ipc_driver_linksys_wvc54gca_8" audio_check_pre="audio_in="
   audio_enable_url="/adm/set_group.cgi?group=AUDIO&amp;audio_in="
   audio_check_url="/adm/get_group.cgi?group=AUDIO"/>
<video_encoder base_url="/adm/set_group.cgi?group=MPEG4"
   bit_rate_url="bit_rate=" frame_rate_url="frame_rate="
   supported_bit_rate="0,1,2,3,4,5,6,7,8,9,10" resolution=""
```

```
      supported_qualities="1,2,3,4,5" frame_rate="" encoding="mpeg4"
      quality_level_url="quality_type=1&amp;quality_level="
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_linksys_wvc54gca_9" supported_framerate="10,20,30"
      resolution_url="resolution="/>
   <video_encoder base_url="/adm/set_group.cgi?group=JPEG" frame_rate=""
      frame_rate_url="frame_rate=" encoding="mjpeg"
      supported_resolutions="160x120,320x240,640x480"
      id="ipc_driver_linksys_wvc54gca_10"
      resolution_url="resolution=" resolution=""
      supported_framerate="10,20,30"/>
   <size id="ipc_driver_linksys_wvc54gca_11" command="1" mjpeg="1"
      width="160" mpeg4="1" height="120"/>
   <size id="ipc_driver_linksys_wvc54gca_12" command="2" mjpeg="2"
      width="320" mpeg4="2" height="240"/>
   <size id="ipc_driver_linksys_wvc54gca_13" command="3" mjpeg="3"
      width="640" mpeg4="3" height="480"/>
</cp7.vns.ipc_driver_linksys_wvc54gca.is_all>
<cp7.vns.o0.is_all level="1" created="1315608566883"
   modified="1315608566883" urgency="none" type="log"
   creator="alerts@portal" text="THIS IS AN ALERT TEST!!!"/>
<cp7.vns.home.is_all created="1315423503988" modified="1315609230662"
   state="stay" type="home" creator="test"/>
<cp7.vns.home_home.is_all state="home" delay="30" trigger="markup"
   created="1315423503996" modified="1315423503996"
   type="home_state" creator="test">
   <trigger id="-1080653838">
      <time time="1800" days="23456"/>
   </trigger>
      <trigger id="-1042217402">
      <time time="0600" days="1234567"/>
   </trigger>
   <script id="home_home_0">
      <tweener..new type="tween" end="${state}" attribute="vns.home.state"
         tick="1000" steps="${delay}" algorithm="countdown"/>
      </script>
</cp7.vns.home_home.is_all>
<cp7.vns.ipc_driver_dlink_dcs5300.is_all upnp_model_number="DCS-5300"
   color="color=" panleft="move=left" setpresetreqauth="false"
   delpreseturl="/setup/ptcamera.cgi" color_color="COLOR"
   created="1315423504699" modified="1315423504699" color_grayscale="B/W"
   name="D-Link DCS-5300" creator="fakeclient" colorurl="/setup/video.cgi"
```

```
      overridesize="size=" setpreseturl="/setup/ptcamera.cgi"
      setpresettext="addpos=" implements="pan,tilt,preset"
      recallpreset="/cgi-bin/recall.cgi?recall="
      controlurl="/cgi-bin/camctrl.cgi?" delpresettext="delpos="
      tiltup="move=up" type="ipc_driver" image="/cgi-bin/video.jpg?"
      panright="move=right" tiltdown="move=down">
      <quality qualityurl="/setup/video.cgi" quality_1="1" quality_2="1"
         quality_3="2" quality_4="2" quality_5="3" quality_6="3" quality_7="4"
         quality_8="4" quality_9="5" id="ipc_driver_dlink_dcs5300_0"
         quality="quan=" quality_10="5"/>
      <size id="ipc_driver_dlink_dcs5300_1" command="1" width="176"
         height="120"/>
      <size id="ipc_driver_dlink_dcs5300_2" command="2" width="352"
         height="240"/>
      <size id="ipc_driver_dlink_dcs5300_3" command="3" width="704"
         height="480"/>
   </cp7.vns.ipc_driver_dlink_dcs5300.is_all>
</fcml>
```

Portal Client UI APIs

7

Communicate with portal server clients

A portal UI client is a web application that runs on the portal server. The web application provides the interface for a device to communicate with the portal server. This chapter describes the FCML messages a portal UI client plug-in can use to communicate wit the portal server.

See *Chapter 7, Portal Client UI APIs* for an example plug-in and a description of the query classes. A web application can use the query classes to communicate with the portal server with minimum coding required.

This chapter covers the following topics:

Communication Entry Points

control points, portal user interfaces (web applications), and external modules are all portal communication entry points. When any of the entry points communicate with the portal server, they address an FCML message request to the applicable portal client.

Each portal client has objects and types. Every object has a type, but not all types have an associated default object. Types let you create a series of objects of a certain type as needed.

Static Objects

Objects can be static or nonstatic. Static means there can be only one instance of the object. For example an `account` object is always static because there can be one, and only one, instance of a customer account for the same person.

Nonstatic means that there can be multiple instances of the object. For example, there could be any number of installed user interfaces on the portal server.

`locations` Client

The `locations` client provides a mechanism to manage location name (address) information. The locations client has a list of all of the states and provinces that can be used.

`state` Type

A static object of type `state` has the following attributes.

Attributes	Data Type	Description
name	String	The name of the location.
Short_Name	String	The short name of the location.

Request

```
<fcml from="ui@portal" to="locations@portal"/>
         <get/>
</fcml>
```

Response

```
<fcml to="ui@portal" from="locations@portal">
   <o1.is_all type="state" name="Alabama" Short_name="AL"/>
   <o1.is_all type="state" name="Arizona" Short_name="AZ"/>
</fcml>
```

cache Client

The cache client provides a mechanism for storing and retrieving data from cache. This client implements the store and recover methods in addition to the common methods.

store method

The store method stores an attribute and value pair in a node. The store method receives that node as an attribute that is then passed into a portal client HTTP session. The name becomes portal_cache_+ name in the session. There is no role restriction.

recover method

The recover method reverses what the store method did. The recover method gets the name and value pair in the HTTP session. The recover method then sends them back as a name and value pair attribute in the returned FCML message. The name is reverted from portal_cache_+ name to name. There is no role restriction.

accounts Client

The accounts client manages user data for each customer in the system.

my_basic_account Object

The static my_basic_account object is type basic_account and has the following attributes, elements, and methods.

Attributes

Attributes	Data Type	Description
id	String	The ID of the object.
cps_owned	Integer	The number of control points that are owned by this account.
cps_shared	Integer	The number of control points that are shared with other accounts.
deliver_high	String	The notification method for messages with a high priority.
deliver_normal	String	The notification method for messages with a normal priority.

Attributes	Data Type	Description
deliver_low	String	The notification method for messages with a low priority.
disabled	Boolean	When True, the account is no longer usable.
locked	Boolean	When True, the customer is locked out of their account. This lockout is usually because the customer tried the wrong password 3 times.
locale	String	The geographical location where the customer is located.
first_name	String	The first name of the customer.
last_name	String	The last name of the customer.
name	String	The nickname of the customer.
service_base_url	String	The URL of the portal server this account is tied to.
storage_used	Integer	The number of megabytes of storage that are used on the portal server.
storage_capacity	Integer	The number of megabytes available to this customer on the portal server.
yahoo_id	String	The Yahoo instant messaging ID of the customer.
email	String	The email address of the customer.
disable_reason	String	The reason the account is disabled.
username	String	The login user name of the customer.
create_date	Date	The date this account was created.
partner	com.fourhomemedia.portal.entity.Partner	The ID of the enterprise partner that this account is under
valid_email	Boolean	True when the email is valid.

Request

```
<fcml from="netrouter@cp4" to="accounts@portal"
                           _debuggerId="043082469163864967>
   <my_basic_account.get/>
</fcml>
```

Response

```
<fcml from="accounts@portal" to="netrouter@cp4"
                           _debuggerId="043082469163864967" >
   <my_basic_account.is_all storage_used="0" disable_reason=""
```

```
        deliver_low="" deliver_normal="email" type="basic_account"
        cps_shared="0" cps_owned="1" id="10" first_name="Joe" u
        sername="16220835" create_date="1313538007000"
        email="joeuser@example.com" storage_capacity="2147483648"
        name="Joe User" last_name="User"
        service_base_url="http://joe.example.com/p/803660/"
        partner="p803660" locked="false" deliver_high="email,sms"
        disabled="false">
     <role id="role39">Client</role>
     </my_basic_account.is_all>
  </fcml>
```

Request

```
  <fcml from="netrouter@cp4" to="accounts@portal"
                              _debuggerId="03365566529657438" >
     <my_basic_account.a10.get/>
  </fcml>
```

Response

```
<fcml from="accounts@portal" to="netrouter@cp4" _debuggerId="03365566529657438" >
   <a10.is_all storage_used="0" disable_reason="" deliver_low=""
      deliver_normal="email" residential="false" type="account"
      cps_shared="0" cps_owned="1" id="10" time_zone="America/Los_Angeles"
      first_name="Joe" username="16220835" create_date="1313538007000"
      email="joeuser@example.com" storage_capacity="2147483648" name="Joe User"
      last_name="User" service_base_url="http://joe.example.com/p/803660/"
      partner="p803660" locked="false" deliver_high="email,sms" disabled="false">
   <phone id="phone14" type="1"/>
   <address id="address10" zip="95124" country_code="US"
         country_name="United States" city="San Jose"/>
   <role id="role39">Client</role>
   </a10.is_all>
</fcml>
```

Elements

`role` Element.

Attributes	Data Type	Description
groupId	String	The ID of the group to which the role belongs.

Methods

`change_username` method.

- **Role restriction**. PartnerAdmin, TechAdmin, Client.
- **Description**. This method changes the user name of the corresponding account. The corresponding account is the account that owns the active client or the control point that sends the message. An email notification is sent to the customer.

  Pass the current password and new user name as attributes to this method with the attribute names `current_password` and `new_username`.

`change_primary_email` method:

- **Role restriction**. Admin, TechAdmin, Client.
- **Description**. This method changes the primary email of the specified account. The target object has to be the ID of the account. The requestor has to be in Admin role or TechAdmin role. An email notification is sent to the customer. Pass the new email address as an attribute with the attribute name `new_email`.

`changePassword` method:

- **Role restriction**. PartnerAdmin, TechAdmin, Client.
- **Description**. This method changes the password for the corresponding account. If the message has the FCML ID of the account, that is the account whose password is changed. If it is specified as `my_account` or `my_basic_account`, the corresponding account password is changed.

  Pass the current password and new password as attributes with the attribute names `current_password` and `new_password`.

`push_mobile_ui` method:

- **Role restriction**. PartnerAdmin, TechAdmin, Client.
- **Description**. This method pushes a message for a mobile device to a mobile phone. Pass the mobile phone ID as an attribute with the attribute name `mobile_id`. If the target object is `my_account` or `my_basic_account`, the implied account is the account that owns the active client or the control point sending the message. If the account ID is the target object of the message, the implied account is the account with the specified ID.

`release_held_messages` method:

- **Role restriction**. PartnerAdmin, TechAdmin, Client.
- **Description**. This method releases all held messages for the implied account. If the target object is the ID of an account, the held messages of that account are released. If the target object is `my_account` or `my_basic_account`, the implied account is the account that owns the active client or control point. The held messages are sent through the mobile phone when the account has one, or through email otherwise. The held messages are removed from the database.

my_account Object

The static my_account object is type account and has the following attributes and elements.

Attributes

Attributes	Data Type	Description
company_name	String	The name of the company where the account owner works.
job-title	String	The job title of the account owner.
residential	Boolean	Whether the account is for a home or a business.

Elements

email Element.

Attributes	Data Type	Description
name	String	The user name of the account owner.
email	String	The email address of the account owner.

mobile Element.

Attributes	Data Type	Description
name	String	The name of the mobile device.
type	Integer	The type of mobile device.
number	String	The access number for the mobile device.
make	String	The make of the mobile device.
model	Integer	The model of the mobile device.
provider	com.fourhomemedia.portal.entity.CellPhoneProvider	The ID of the mobile phone service provider.

phone Element.

Attributes	Data Type	Description
name	String	The name of the phone.
type	Integer	The type of phone.
number	String	The phone number.
extension	String	The extension number (if any).

address Element.

Attributes	Data Type	Description
city	String	The city of the account owner.
country_code	Integer	The country code of the account owner. See *Appendix A, Region, Country, and ISP Codes*.
country_name	String	The country of the account owner.
state	String	The state of the account owner.
street1	String	The street address of the account owner.
street2	String	A second line for the street address.
zip	String	The zip code for the street address.

my_locations Object

The nonstatic my_locations objects has the location element with the following attributes.

Attributes	Data Type	Description
Name	String	The name of the location.
zip	String	The zip code for the location.
city	String	The location city.
default	Boolean	Whether this location is the default location (1) or not (0).

my_preferences Object

The nonstatic my_preferences object is type my_preferences and has the following element and method. Application developers can create their own set of configuration varialbles for the customer. The variables can be any name and any value that the application developer chooses. These configuration variables are stored in the my_preferences object.

Element

preferences Element.

Attributes	Data Type	Description
name	String	The name of the configuration preference.
value	String	The value of the configuration preference.

Method

reset method.

Role restriction. PartnerAdmin, TechAdmin, Client.

Description. Reset the account preference of the account to which the control point or the active client belongs. The control point or the active client is the sender of the message.

`customer` Type

Objects of type `customer` are customer account objects. Customer account objects have three attributes of type `String`, which are low, medium, and high. These attributes indicate the delivery method for notifications with those respective urgencies.

`time_zones` Object

The static `time_zones` object is type `time_zone` and has the following element.

`time_zone` Element.

Attributes	Data Type	Description
timezone_id	String	The time zone ID.
gmt_offset	String	The number of hours the time zone is from Greenwich Mean Time.
name	String	The name of the time zone.

`countries` Object

The static `countries` object is type `country` and has the following element.

`country` Element.

Attributes	Data Type	Description
id	String	The country ID.
code	String	The country code. See *Appendix A, Region, Country, and ISP Codes*.
name	String	The name of the country.

`account_contact` Type

A nonstatic object of type `account_contact` has the following attributes.

Attributes	Data Type	Description
name	String	The full name of the contact person.

Attributes	Data Type	Description
email	String	The email address for the contact person.
account	com.fourhomemedia.portal.entity.Account	The ID of the account this contact is associated with.

`installer_note` Type

A nonstatic object of type `installer_note` has the following attributes. When somebody is doing an installation of the hardware and software, they can create notes pertaining to what they did.

Attributes	Data Type	Description
created	.Date	The date the note was entered.
account	com.fourhomemedia.portal.entity.Account	The ID of the account to which this note pertains.
installer	com.fourhomemedia.portal.entity.Account	The ID of the account of the installer.
installer_name	String	The name of the installer.
note	String	The text of the note.
read	Boolean	When True, the note has been read.

`account_message` Type

A nonstatic object of type `account_message` has the following attributes.

Attributes	Data Type	Description
created	Date	The date the message was created.
to_account_name	String	The account that this message is directed to.
from_account_name	String	The account that this message is from.
to_account	com.fourhomemedia.portal.entity.Account	The ID of the receiving account.
from_account	com.fourhomemedia.portal.entity.Account	The ID of the sending account.
message	String	The text of the message.

Attributes	Data Type	Description
old	Boolean	When True, this message has been read.
partner	com.fourhomemedia.portal.entity.Partner	The ID of the enterprise partner the accounts are under.

`activatable_control_points` Object

The nonstatic `activatable_control_points` object has the following attributes.

Attributes	Data Type	Description
model	String	Hardware model SKU
date	String	Date the control point first contacted the server
image	String	A URL to an image of the device

`mobile_models` Object

The nonstatic `activatable_control_points` object has the following attributes. This object contains descri0tion information about the cell phone.

Attributes	Data Type	Description
ID	Integer	The ID of the mobile device model.
model	String	The model of the mobile device.
name	String	The friendly name of the mobile device model.

`mobile_makes` Object

The nonstatic `activatable_control_points` object has the following attributes.

Attributes	Data Type	Description
ID	Integer	The ID of the mobile device manufacturer.
make	String	The manufacturer of the mobile device.
name	String	The friendly name of the cell mobile device manufacturer.

mobile_operators Object

The nonstatic activatable_control_points object has the following attributes.

Attributes	Data Type	Description
name	String	Name of the operator
max_length	String	Maximum length of an SMS message for this carrier

links Client

The links client provides a mechanism to retrieve the URLs for various applications or resources within the ecosystem. Each link object has the following attributes.

Attributes	Data Type	Description
has_active_cps		Indicates whether a control point is connected to that link.
url		The URL to retrieve the resource.
id		The link object ID.

control Object

The nonstatic media object is type link and has the links client attributes.

setup Object

The nonstatic media object is type link and has the links client attributes.

help Object

The nonstatic media object is type link and has the links client attributes.

users_guide Object

The nonstatic media object is type link and has the links client attributes.

accounts object

The nonstatic media object is type link and has the links client attributes.

client_logs Client

The client_logs client lets an application log, issue, or query notification events.

log Object

The log object is a log message. It has the following attributes and methods.

Attributes	Data Type	Description
id	Integer	The log message ID.
level_value	Integer	The level of the logged event. The values are the Java standard logging levels: SEVERE (highest), WARNING, INFO, CONFIG, FINE, FINER, FINEST (lowest).
time_stamp	Date	The time the event was logged.
client_id	String	The ID of the client that logged the event.
control_point	Integer	The control point domain that logged the event.
text	String	The description of the event.
urgency	String	The urgency of the event. Values can be none, low, normal, or high.
viewed	Boolean	A value of True means the log has been viewed.

countUnviewed Method

In this example, there is a log table that stores some log messages. Each log message, is marked either unviewd or viewed. The countUnviewd method counts the number of log messages that are marked unviewed. The response message returns the number of unviewed messages. In this example, the number of unviewed messages is 8.

Request:

```
<fcml from="ui446@portal" to="logs@portal">
   <countUnviewed />
</fcml>
```

Response:

```
<fcml from="log@portal" to="ui446@portal" >
   <is_all numOfUnread="8"/>
</fcml>
```

set Method

Use the `set` method to mark a message as either `viewed` or `unviewed`. The `fcmlid` of the log message is `log2`.

Request:

```
<fcml from="ui446@portal" to="logs@portal">
   <log2.set viewed="true" />
</fcml>
```

Response:

```
<fcml from="logs@portal" to="ui446@portal" >
   <log2.is viewed="true" type="log"/>
</fcml>
```

getSomeByAccount Method

Use the `getSomeByAccount` method to get the messages belonging to a client account. This method gets messages by page. The `StartIndex` attribute is the index of first message and the `unmOfLogs` attribute is the page size. This method returns messages that are ordered by time, with the most recent message first.

Request:

```
<fcml from="ui446@portal" to="logs@portal">
   <getSomeByAccount startIndex="2" numOfLogs="4"/>
</fcml>
```

Response:

```
<fcml from="log@portal" to="ui446@portal" >
   <is_all>
      <log id="11" text="This is text" controlPoint="cp4" level="1"
      timeStamp="2011-12-12 00:00:00.0" viewed="true" clientId="ui132"
      urgency="warning"/>
      <log id="10" text="This is text" controlPoint="cp4" level="1"
      timeStamp="2011-12-12 00:00:00.0" viewed="false" clientId="ui131"
      urgency="warning"/>
   </is_all>
</fcml>
```

getAllByAccount Method

Use the getAllByAccount method to get all the messages that belong to a client account.

Request:

```
<fcml from=\"ui446@portal\" to=\"logs@portal\">
   <getAllByAccount />
</fcml>"
```

Response:

```
<fcml from="log@portal" to="ui446@portal" >
   <is_all>
      <log id="13" text="This is text" controlPoint="cp4" level="1"
      timeStamp="2011-12-12 00:00:00.0" viewed="false" clientId="ui134"
      urgency="warning"/>
      <log id="12" text="This is text" controlPoint="cp4" level="1"
      timeStamp="2011-12-12 00:00:00.0" viewed="true" clientId="ui133"
      urgency="warning"/>
      <log id="11" text="This is text" controlPoint="cp4" level="1"
      timeStamp="2011-12-12 00:00:00.0" viewed="true" clientId="ui132"
      urgency="warning"/>
   </is_all>
</fcml>
```

delete Method

Use the delete method to delete the message with the specified fcmlid.

Request:

```
<fcml from="ui446@portal" to="logs@portal">
   <log2.delete/>
</fcml>
```

Response:

```
<fcml from="logs@portal" to="ui446@portal" >
   <portal.logs.log2.is_deleted/>
</fcml>
```

get Method, Sample 1

Use the get method to retrieve all of the messages with the specified fcmlid. In this example, log2 is the fcmlId, where log is the prefix and 2 is the message ID.

Request:

```
<fcml from="ui446@portal" to="logs@portal">
   <log2.get/>
</fcml>
```

Response:

```
<fcml from="logs@portal" to="ui446@portal" >
   <log2.is_all id="2" text="This is text" viewed="true" client_id="ui123"
   type="log" time_stamp="1323676800000"
   control_point="com.fourhomemedia.portal.entity.ControlPoint[id=4]"
   urgency="warning"/>
</fcml>
```

get Method, Sample 2

This get method example gets all the logs that belong to the account of the logged-in customer.

id attribute. The database id of the message,

text attribute. The message text.

level_value attribute. The level of the message. The values are the Java standard logging levels: SEVERE (highest), WARNING, INFO, CONFIG, FINE, FINER, FINEST (lowest).

viewed attribute. A flag to indicate whether the message has been viewed or not.

client_id attribute. The id of the client that is logging the message.

type attribute. The type of the object. In this case, it is always "log".

time_stamp attribute. The time the message is logged.

control_point attribute. The ID of the control point.

urgency attribute. The message urgency. Values can be info, warning, fatal, and so on.

Request:

```
<fcml from="ui447@portal" to="logs@portal">
   <get path="log"/>
</fcml>
```

Response:

```
<fcml from="logs@portal" to="ui447@portal" >
   <log1.is_all id="1" text="this is text" viewed="false" client_id="ui256"
```

```
   type="log" time_stamp="1323417600000"
   control_point="com.fourhomemedia.portal.entity.ControlPoint[id=4]"/>
   <log4.is_all id="4" text="This is text" level_value="1" viewed="true"
   client_id="ui125" type="log" time_stamp="1323676800000"
   control_point="com.fourhomemedia.portal.entity.ControlPoint[id=4]"
   urgency="warning"/>
   <log5.is_all id="5" text="This is text" level_value="1" viewed="false"
   client_id="ui126" type="log" time_stamp="1323676800000"
   control_point="com.fourhomemedia.portal.entity.ControlPoint[id=4]"
   urgency="warning"/><log6.is_all id="6" text="This is text" level_value="1"
   viewed="false" client_id="ui127" type="log" time_stamp="1323676800000"
   control_point="com.fourhomemedia.portal.entity.ControlPoint[id=4]"
   urgency="warning"/><log7.is_all id="7" text="This is text" level_value="1"
   viewed="false" client_id="ui128" type="log" time_stamp="1323676800000"
   control_point="com.fourhomemedia.portal.entity.ControlPoint[id=4]"
   urgency="warning"/>
</fcml>
```

client_subscription Client

The client_subscription client has the id attribute of type Integer and the subscribe and unsubscribe methods. Use these methods in your client plug-ins to form FCML messages that subscribe to and unsubscribe from the FCML messages of other clients.

Subscribe Sample

This sample shows how to request a subscription and the response. The response returns subscription12, which is the fcmlid of the subscription that can be used later to unsubscribe.

The path attribute is specified in XPath format. The path attribute supports any string that follows the xpath syntax. See *http://www.w3schools.com/xpath/xpath_syntax.asp* for information on XPath syntax.

> ***Note:*** *When you form a value for the path attribute, make sure the beginning double forward slashes (//) are omitted.*

Request:

```
<fcml from="ui402@portal" to="subscription@portal">
   <subscribe path="accounts.my_basic_account[@username='16220835']"/>
</fcml>
```

Response:

```
<fcml to="ui402@portal" from="subscription@portal">
   <subscription12.is />
</fcml>
```

Unsubscribe Sample

A client plug-in sends an unsubscribe FCML message to remove the subscription. A client plug-in can unsubscribe only from its own subscriptions. This sample uses the `fcmlid` of `subscription12` to unsubscribe from its own subscription. If a client tries to unsubscribe from a subscription that belongs to another client, the request is declined and the response contains `unsubscribed ="false"`.

Request:

```
<fcml from="ui402@portal" to="subscription@portal">
   <subscription12.unsubscribe type="subscription"/>
</fcml>
```

Response:

```
<fcml to="ui402@portal" from="subscription@portal">
   <subscription12.isDeleted unsubscribed="true"/>
</fcml>
```

Control Point Plug-In Example

8

An control point plug-in with code

This chapter describes the code for a Hello World! client plug-in that runs on and extends the functionality of a control point.

This chapter covers the following topics:

Development Environment Setup

In addition to the setup described in *Chapter 2, How to Set Up, Package, & Deploy*, get the following files to develop a Control Point client plug-in:

- fcsample.zip file
- NETGEAR Control Point libraries.

The fcsample.zip file contains a simple Hello World stand-alone client program. The application comes with an Ant script, and is organized into the following folders:

- The build.xml Ant script
- The src folder that contains the source code under the following directory structure:

  ```
  ~\src\netgear\sample\FcmlHelloWorld.java
  ~\src\netgear\sample\HelloWorldService.java
  ```

  The build.xml file is an Ant script that contains the NETGEAR Control Point project and the build targets. By default all helloworld tasks are built. The helloworld target calls the JAR files in the lib directory, described next.

  The Hello World sample application uses the felix.jar and commons.jar libraries. The JAR files contain the class files and other information for executing a client application in NETGEAR Control Point. See *NETGEAR Control Point Libraries* on page 142 for a description of felix.jar and commons.jar. *NETGEAR Control Point Libraries* on page 142 also has a full list of JAR files for a larger application.

NETGEAR Control Point Libraries

A full NETGEAR Control Point project consists of the following files and directories. A listing at the root of the core source hierarchy displays the following directories:

- **build.xml**. The Ant build script.
- **lib**. The root directory for the library code, such as commons.jar.
- **src**. The root source directory for all non-plug-in clients.
- **plug-ins**. The root source directory for all plug-in clients.
- **init**. The directory that contains the initialization scripts for the VNS client.
- **wwwroot**. The directory for the basic index.html file served by the core code.
- **felix.jar**. This JAR file contains the org and META_INF directories, and a default.properties file. The org directory has the `apache` and `osgi` classes. The Java platform reads the META-INF subdirectories to configure applications, extensions, class loaders, and services. The default.properties file contains framework configuration properties and package export properties for the NETGEAR Control Point platform.
- **commons.jar**. This JAR file contains the com and META-INF directories. The com directory contains the `fourhome` classes under the following directory structure:

  ```
  ~\com\fourhome\commons\*.class
  ```

Create the Plug-In

Before you begin, decide the functionality you want your client plug-in to have. Think about the types of communications required with the other services. There are two types of client plug-ins: agents and drivers. Agents perform device-independent services, and drivers control devices. Every client plug-in has a name so the system and services can identify it. By convention Smart Network client plug-in names are lowercase. For this example, the Hello World client plug-in is an agent named `hello`.

`HelloWorldService.java` Source Code

The `HelloWorldService` class extends the `PluginCoreClient` class that implements the `getServiceName` and `getServiceType` methods. The extended class is installed on the control point as the starting point for the plug-in.

Complete `HellowWorldService.java` Source Code Listing

A brief explanation follows this source code listing.

```
/*
 * HelloWorldService.java
 * Copyright (c) 2011, NETGEAR, Inc. All rights reserved.
 */
package com.netgear.sample;
import java.io.IOException;
import com.fourhome.commons.ClientPipe;
import com.fourhome.commons.CoreService;
import com.fourhome.commons.Fcml;
import com.fourhome.commons.Message;
import com.fourhome.commons.PluginCoreClient;

/**
 * HelloWorldService extends PluginCoreClient. PluginCoreClient is the
 * superclass that plug-ins extend because it represents the connection
 * point to the NETGEAR Control Point environment. In this example,
 * HelloWorldService starts one OSGi service.
 */
public class HelloWorldService extends PluginCoreClient {
   public static final String SERVICE_NAME = "hello";
   public static final String SERVICE_TYPE = CoreService.Type.AGENT.toString();

   public String getServiceName() {
      return SERVICE_NAME;
   }
```

```
   public String getServiceType() {
      return SERVICE_TYPE;
   }

   /**
    * A plug-in handles messages that are sent to it from other services.
    * @param msg The message.
    * @throws IOException Thrown upon a bad message (bad fcml).
    */
   public void handleMessage(Message msg) throws IOException {
      ClientPipe pipe = getClientPipe();
      FcmlHelloWorld response = new FcmlHelloWorld (msg, pipe);
      Fcml resp = response.getResponse();
      if(resp != null)
         pipe.sendMessage(resp);
      resp = response.getBroadcastResponse();
      if(resp != null)
         pipe.sendMessage(resp);
      }

   public void run() {
      if (isRunning()) {
         startReceiving();
      }
   }
}
```

Default Constructor

The client plug-in class is instantiated with its default constructor. Do not create any other constructors because they are not used. The `HelloWorldService.java` class does not have a constructor, which means the default constructor is called.

`handleMessage` Method

The `handleMessage` method receives a message, which is a request for information from a client (a `get` request). In response, this method instantiates a pipe between itself and the client. The method then instantiates an `FCMLHelloWorld` object to process the message, and returns a response with the requested information. Finally, the `handleMessage` method returns the response to the requestor through the pipe of the requestor.

```
public void handleMessage(Message msg) throws IOException {
   ClientPipe pipe = getClientPipe();
```

```
   FcmlHelloWorld response = new FcmlHelloWorld (msg, pipe);
   Fcml resp = response.getResponse();
   if(resp != null)
      pipe.sendMessage(resp);
      resp = response.getBroadcastResponse();
   if(resp != null)
      pipe.sendMessage(resp);
   }
```

Implement the Run Method

The `PluginCoreClient` parent class implements `runnable`, which has a `run` method to start the plug-in within its own thread. An implementation for the `run` method is added to the `HelloWorldService` class as follows:

```
public void run() {
   if (isRunning()) {
      startReceiving();
   }
}
```

`FcmlHelloWorld.java` Source Code

The `FcmlHelloWorld` class responds to `get` requests from `HelloWorldService` objects. This class extends the `FcmlExecutor` class, which is the base class for all classes that handle or run FCML messages.

Complete `FcmlHellowWorld.java` Source Code Listing

A brief explanation follows this source code listing.

```
/*
 * FcmlHelloWorld.java
 * Copyright (c) 2011, NETGEAR, Inc. All rights reserved.
 */

package com.netgear.sample;
import java.io.IOException;
import java.util.Random;
import com.fourhome.commons.ClientId;
import com.fourhome.commons.ClientPipe;
import com.fourhome.commons.FcmlExecutor;
import com.fourhome.commons.FluidObject;
import com.fourhome.commons.Message;
import com.fourhome.commons.MethodId;
```

```
/**
 * FcmlHelloWorld extends FcmlExecutor. FcmlExecutor is the base class
 * for all classes that handle or run FCML messages.
 */
public class FcmlHelloWorld extends FcmlExecutor {

//Create a Logger object
   private static final Logger logger =
      LoggerFactory.getLogger(FcmlHelloWorld.class);

   private ClientId myId;
   private ClientPipe pipe;

   static final String OBJECT_RANDOM_NUMBER = "randomNumber";
   /**
    * Handle a get request from a client process.
    * @param myId The ID of the client making the request.
    * @param node The FCML node that contains the request message.
    * @return True if the get method was handled.
    */

   protected boolean handleGet(ClientId myId, FluidObject node) {
      if ( !super.handleGet(myId, node) ) {
         MethodId id = node.getId();
         String nodeObject = id.getObject();

      if ((id == null) || (nodeObject == null)) {
         handleGetAll(node);
      else if (OBJECT_RANDOM_NUMBER.equals (id.getObject())) {
         FluidObject ret = new FluidObject
               (new MethodId (pipe.getId().getDomain()
            + "." pipe.getId().getClient()
            + "." + OBJECT_RANDOM_NUMBER
            + ".FcmlExecutor.IS));
      {
         Random randomGen = new Random();
         randomGen.setSeed(System.currentTimeMillis());
         ret.putAttribute(OBJECT_RANDOM_NUMBER, Integer.toHexString
            (randomGen.nextInt ()));
      }
      ret.putAttribute(FcmlExecutor.ATTRIBUTE_TYPE, "sampleProgram");
      applySelectAndRespond (myId, null, ret, node);
```

```
    }
    else {
       //If the handleGet method fails, log error message
       logger.warn("Received unrecognized method: " + id.getObject ());
       return false;
    }
    return true;
}

/**
* Handle a get method without an object
* @param node The fluid object of the method
*/
    private void handleGetAll(FluidObject node) {
       //  look for incoming attributes
       String whom = node.getAttributeValue("whom");

       FluidObject ret = new FluidObject (new MethodId (pipe.getId().getDomain()
                                          + "." + pipe.getId().getClient()
                                          + "." + FcmlExecutor.IS_ALL));
       //  add the outgoing attributes
       if (whom != null)
          ret.putAttribute("text", "hello " + whom + "!");
       else
          ret.putAttribute("text", "hello world!");
          ret.putAttribute(FcmlExecutor.ATTRIBUTE_TYPE, "sampleProgram");
          applySelectAndRespond (myId, null, ret, node);
       }

    /**
     * Construct a new FcmlExecutor based on the FCML message passed in.
     * @param node The fcml message
     * @throws IOException Thrown if message is not actually an fcml message
     */
    public FcmlHelloWorld (Message message, ClientPipe pipe) throws IOException{
       super(message);
       this.myId = pipe.getId();
       this.pipe = pipe;
       process(myId);
    }
}
```

Constructor

The `FCMLHelloWorld` constructor creates an instance that is based on the FCML message that is passed in.

```
public FcmlHelloWorld (Message message, ClientPipe pipe) throws IOException {
   super(message);
   this.myId = pipe.getId();
   this.pipe = pipe;
   process(myId);
}
```

`handleGet` and `handleGetAll` Methods

The `handleGet` method is called behind the scenes when `response.getResponse()` is called in the `HelloWorldService` object. The `handleGet` method checks for the client and node (message) ID. If no client ID is found, `handleGet` calls the `handleGetAll` method with only the node ID. The `handleGetAll` method returns all attributes and their values. Otherwise, the `handleGet` method processes the message by returning information on the attribute or attributes for which value information is requested. In this example the method returns the `Object_Random_Number` attribute with a random number for the attribute value.

`handleGet` Error Logging

The `FcmlHelloWorld` class includes error logging. A call to the `LoggerFactory.getLogger` method creates a `logger` object. The `handleGet` method logs error text to the `logger` object when an unrecognized node (message) ID is passed into the `if` statement.

This example shows how to handle an incoming request. See *Logs and Alerts* on page 29 for information on how to format an incoming request.

Portal UI Plug-In Example

9

A portal UI plug-in with code

This chapter describes the process and APIs for creating a Web-based User Interface plug-in. A portal UI plug-in is a web application that runs on the portal server. Customers connect to the portal UI from their smart home device to communicate with Smart Network and its external services.

This chapter covers the following topics:

- *Install Google Web Toolkit (GWT)*
- *Add or Update NetgearGWTAppHelper*
- *Sample Web Application*

See *Chapter 7, Portal Client UI APIs*, for information on the portal clients and objects that the Query classes interface with.

Install Google Web Toolkit (GWT)

You need the latest version of the Google Web Toolkit (GWT) plug-in for Eclipse installed.

➢ **To get GWT:**

- From within Eclipse, go to **Help > Install New Software**.

Or

- On the Web, go to *http://code.google.com/eclipse/docs/download.html* to download and GWT.

  See *http://code.google.com/eclipse/docs/getting_started.html* for additional information.

Add or Update NetgearGWTAppHelper

NetgearGWTAppHelper (Helper) is a GWT helper module. Add it to your GWT project.

Add the Helper Project to Eclipse

The Helper project is available on the developer web site.

➢ **To add the Helper Project to Eclipse:**

1. Extract the Helper project onto your computer.
2. Start **Eclipse**
3. Inside Eclipse, select **file > import.**
4. In the Import dialog box, click the plus sign (**+**) next to General.
5. Select **Existing Projects into Workspace** in the General list and click **Next.**
6. In the Import Projects dialog box, make sure that the **Select root directory** radio button is selected and click **Browse...**
7. Select the **NetgearGWTAppHelper**. project.
8. Click **OK** and then **Finish**.

   If Eclipse shows errors after the import, select **Project > Clean** to refresh the project.

   Note: *Make sure that there are no errors before you proceed.*

About NetgearGWTAppHelper

The Helper project communicates with the Portal server and the control points owned by the customer. WIth the Helper project, you can concentrate on your application (UI) and business logic without having to write any communications code. This model separates the message communications code from the core application code.

The Helper project has the following four types of classes that are described in the following sections:

- Queriy (Request) classes
- `Cache` class
- `FluidObject` (Response) class
- `FCML` class

Query (Request) Classes

These Query (request) helper classes facilitate sending FCML requests to the portal server.

- `ClientObjectQuery` class
- `ClientQuery` class
- `CPQuery` class

`ClientObjectQuery` Class

Use this query to request information from a portal object. You provide the portal client and object or the list of portal objects under the client that you want to query.

The following code sample creates a `ClientObjectQuery` object and sends the query object to `accounts@portal`. The query requests the account information for `my_basic_account`.

```
   ClientObjectQuery query = new ClientObjectQuery("accounts@portal", new
String[]{"my_basic_account "});
      query.addQueryListener(new QueryListener() {
         @Override
         public void onResult(Query response) {
            ArrayList<FluidObject> results = response.getResults();
            // your desired code goes here
         }
      });
      query.execute();
```

The FCML request sent to the portal server is the following:

```
   <fcml from="uiXY@portal" to="accounts@portal">
      <my_basic_account.get />
   </fcml>
```

ClientQuery Class

Use this query to request information from a portal client when you only know the portal client and nothing else. The following query requests information from router@portal with a get message:

```
ClientQuery query = new ClientQuery(router);
query.addQueryListener(this);
query.execute();
```

The FCML request sent to the portal server is the following:

```
<fcml from="ui@XYportal" to="router@portal" >
   <get />
<fcml>
```

CPQuery Class

Use this query to request infromation from a portal client when the request is a custom message. The following code sends `my_message` to the client (service) `ntgr_trafficmeter` at control point `cp2`.

```
CPQuery query = new CPQuery("ntgr_trafficmeter@cp2", "my_message");
query.addQueryListener(this);
query.execute();
```

The FCML request sent to the portal server is the following:

```
<fcml from="uiXY@portal" to="ntgr_trafficmeter@cp2">
 <my_message />
</fcml>
```

Cache Class

The `Cache` class maintains a local cache of the responses that are received by `FluidObject` objects.

`FluidObject` (Response) Class

The `FluidObject` class is a response class that parses and maintains responses. The developer does not have handle FCML parsing.

FCML Class

The `FCML` class establishes the communication entry point that enables your code to start portal server communications. This class also handles cross-domain FCML communications.

Sample Web Application

Follow these procedures to set up Eclipse for the sample web application.

➢ **To create a project:**

1. Inside Eclipse, select **File > New > Web Application Project**.
2. In the Project Name field, type **MyInfo**.
3. In the Package field, type **com.netgear.myinfo**.
4. n the New Web Application dialog box under Google SDKS, deselect **Use Google App Engine**, if it is selected. Make sure that the Use Google Web Toolkit is the only option selected.

➢ **To set up the sample web application:**

1. In Eclipse under MyInfoApp, expand **src**.
2. Under src, right click **com.netgear.myinfo** and select **New > Google**.
3. In the Select a Wizard dialog box, expand **Google Web Toolkit**, and select **Module**.
4. Click **Next**.
5. In the Name field of the Entry Point Class dialog box, type **MyInfoApp**.
6. Click **Finish**.

   Eclipse creates the `MyInfoApp.gwt.xml` file is created and puts it under the `com.netgear.myinfo package`. Eclipse also creates `com.netgear.myinfo.client package` and places it under `src`.

➢ **To create the MyInfoApp entry point class:**

1. In Eclipse, right click the `com.netgear.myinfo.client` package and select **New > Other**.
2. In the Select a wizard dialog box, expand `Google Web Toolkit` and select **Entry Point Class**.
3. In the Name field of the Entry Point Class dialog box, type **MyInfoApp** and click **Finish**.
4. Make sure the MyInfoApp.java, MyInfoApp.html, and web.xml files have the correct contents. The correct contents are shown in *`MyInfoApp` Source Code* on page 154, *MyInfoApp.html* on page 156, and *web.xml* on page 156.

➢ **To test MyInfoApp:**

1. Right click the MyInfoApp project and select **Google > GWT compile**.
2. In the Compile dialog box, choose the outpust style for the code. Obfuscated is the default and means that no programs can read the compiled code. Pretty is readable to a human and detailed adds to pretty with more details such as verbose variable names.
3. Package and deploy the application as described in *Chapter 9, Portal UI Plug-In Example*.

MyInfoApp Source Code

The following sample web applications displays two text boxes that contain the name and email of the customer. You can get the MyInfoApp source code on the developer web site..

In the onModuleLoad method at line 27, the call to FCML.init takes the string 1234567890 for the APP_KEY parameter. The string myInfoInitCallback is the application callback for initialization and authorization. See the exportMethods method on line 58, to see how the callback is used.

Line 55 has a JavaScript Native Interface (JSNI) call to the myInfoInitCallback method to create the callback. JSNI calls the callback after your application is initialized and receives authorization.

```
1  package com.myinfo.account.client;
2
3  import java.util.ArrayList;
4  import com.google.gwt.core.client.EntryPoint;
5  import com.google.gwt.event.dom.client.ClickEvent;
6  import com.google.gwt.event.dom.client.ClickHandler;
7  import com.google.gwt.user.client.Window;
8  import com.google.gwt.user.client.ui.Button;
9  import com.google.gwt.user.client.ui.Label;
10 import com.google.gwt.user.client.ui.RootPanel;
11 import com.google.gwt.user.client.ui.TextBox;
12 import com.netgear.appdev.ClientObjectQuery;
13 import com.netgear.appdev.FCML;
14 import com.netgear.appdev.FluidObject;
15 import com.netgear.appdev.Query;
16 import com.netgear.appdev.QueryListener;
17
18 /**
19  * Entry point classes define <code>onModuleLoad()</code>.
20  */
21 public class MyInfoApp implements EntryPoint {
22    TextBox nametext;
23    TextBox emailText;
24    /**
25     * This is the entry point method.
26     */
27    public void onModuleLoad() {
28       FCML.init("1234567890", "myInfoInitCallback");
29       exportMethods(this);
30       loadUI();
31    }
```

```
32
33    private void loadUI() {
34       Label namelbl = new Label("Name : ");
35       nametext = new TextBox();
36       Label emaillbl = new Label("Email : ");
37       emailText = new TextBox();
38       Button edit = new Button("Edit");
39       edit.addClickHandler(new ClickHandler() {
40
41          @Override
42          public void onClick(ClickEvent event) {
43             Window.alert("hey i am clicked");
44          }
45       });
46       RootPanel.get().add(namelbl);
47       RootPanel.get().add(nametext);
48       RootPanel.get().add(emaillbl);
49       RootPanel.get().add(emailText);
50       RootPanel.get().add(edit);
51    }
52
53
54    public native void exportMethods(MyInfoApp instance) /*-{
55       $wnd['myInfoInitCallback'] = function(msg){
56          return
instance.@com.myinfo.account.client.MyInfoApp::initDone()();
57       }
58    }-*/;
59
60    public void initDone(){
61       ClientObjectQuery clientObjectQuery = new
ClientObjectQuery("accounts", new String[]{"my_basic_account "});
62       clientObjectQuery.addQueryListener(new QueryListener() {
63
64          @Override
65          public void onResult(Query query) {
66             ArrayList<FluidObject> results = query.getResults();
67             for(FluidObject result : results)
68                populateNameAndEmail(result);
69          }
70       });
71       clientObjectQuery.execute();
```

```
72    }
73
74    protected void populateNameAndEmail(FluidObject resultObj) {
75       nametext.setText(resultObj.attributes.get("first_name"));
76       emailText.setText(resultObj.attributes.get("email"));
77    }
78 }
```

MyInfoApp.html

```
<!doctype html>
<html>
   <head>
      <meta http-equiv="content-type" content="text/html; charset=UTF-8">
      <title>MyInfoAPp</title>
      <script type="text/javascript" language="javascript" src=
"com.netgear.myinfo.MyInfoApp/com.netgear.myinfo.MyInfoApp.nocache.js"> </script>
   </head>

   <body>
      <iframe src="javascript:''" id="__gwt_historyFrame" tabIndex='-1' style=
"position:absolute;width:0;height:0;border:0"></iframe>
   </body>
</html>
```

web.xml

Make sure the three blue highlighted lines before the end are present.

```
<?xml version="1.0" encoding="utf-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">
   <!-- TODO: Add <servlet> tags for each servlet here. -->
   <!-- TODO: Add <servlet-mapping> tags for each <servlet> here. -->
   <!-- TODO: Optionally add a <welcome-file-list> tag to display a welcome file.
-->
<welcome-file-list>
   <welcome-file>MyInfoApp.html</welcome-file>
</welcome-file-list>
</web-app>
```

Region, Country, and ISP Codes

A

A quick reference guide

This appendix covers the following topics:

Region Values

Region	NewRegion Value
Africa	ZA
Asia	SG
Australia	AU
Canada	CA
Europe	GB
Israel	IL
Japan	JP
Korea	KR2
Mexico	MX
South America	UY
United States	US

Country and ISP Codes

Use the country and ISP codes to set the `NewCountry` and `NewISPName` attribute values. This list is organized alphabetically by country with the code first. The ISP codes for each country have 3 parameters. The second parameter is the ISP name, and the third parameter is the ISP code.

For example, to set the ISP to Optus Internet in Australia, set `NewCountry` to `AU` and `NewISPName` to `1`, as follows:

```
<fcml from="netrouter" to="c2@d2">
   <WAN3GInterfaceConfig.Set3Config NewCountry= "AU", NewISPName="1"
   _sessionId="1743d38c47627b68 />
</fcml>
```

AU (Australia)

{ "", "Optus (internet)", "1" },

{ "", "Optus (preconnect)", "136" },

{ "", "Optus (connect)", "2" },

{ "", "Telstra", "3" },

{ "", "Telstra Turbo21 (internet)", "105" },

{ "", "Telstra Turbo21 (pcpack)", "106" },

{ "", "Telstra Turbo21 (datapack)", "107" },

{ "", "BigPond", "4" },

{ "", "3", "5" },

{ "", "3 (prepaid)", "6" },

{ "", "Virgin", "7" },

{ "", "Vodafone (prepay)", "137" },

{ "", "Vodafone", "8" },

{ "", "Dodo", "10" },

{ "", "Primus", "11" },

{ "", "Blink (postpaid)", "108" },

{ "", "iiNet", "110"} ,

{ "", "Westnet", "111" }

AT (Austria)

{ "", "A1", "112" }

{ "", "T-Mobile", "113" }

{ "", "Orange", "114" }

{ "", "3", "115" }

{ "", "Tele.Ring A", "116" }

{ "", "Tele.Ring B", "117" }

{ "", "Yesss", "177" }

BE (Belgium)

{ "", "BASE", "13" }

{ "", "Mobistar (personal)", "14" }

{ "", "Mobistar (business)", "15" }

{ "", "Orange", "16" }

{ "", "Proximus", "17" }

BR (Brazil)

{ "", "Claro", "18" }

{ "", "Oi", "19" }

{ "", "Oi (WAP)", "20" }

{ "", "TIM", "21" }

{ "", "Velox", "22" }

{ "", "Vivo", "173" }

{ "", "CTBC", "174" }

CL (Chile)

{ "", "Claro", "23" }

{ "", "Entel PCS", "24" }

{ "", "Movistar", "25" }

{ "", "Telefonica (Movil)", "26" }

CZ (Czech)

{ "", "Telefonica O2", "138 }

{ "", "T-Mobile Czech", "139 }

FI (Finland)

{ "", "Dna", "27" }

{ "", "Elisa (Kolumbus)", "28" }

{ "", "Saunalahti", "29" }

{ "", "Sonera", "30" }

{ "", "Song", "31" }

FR (France)

{ "", "Bouygues", "118" }

{ "", "Bouygues", "153" }

{ "", "Orange - Internet Everywhere", "119" }

{ "", "Orange - Business Everywhere", "120" }

{ "", "SFR", "154"}

DE (Germany)

{ "", "O2 (time-based plans)", "34" }

{ "", "O2 (volume-based plans)", "35" }

{ "", "tagesflat", "36" }

{ "", "T-Mobile D1", "37" }

{ "", "Vodafone", "38" }

HK (Hong Kong)

{ "", "CSL", "39" }

{ "", "New World", "40" }

{ "", "Orange", "41" }

{ "", "People", "42" }

{ "", "SmarTone", "43" }

{ "", "Sunday", "44" }

{ "", "3", "45" }

{ "", "PCCW", "176" }

HU (Hungary)

{ "", "T-Mobile", "155"}

IN (India)

{ "", "MTNL Mumbai", "156" }

ID (Indonesia)

{"", "Indosat", "157"}

IT (Italy)

{ "", "Postemobile", "46" }

{ "", "3", "47" }

{ "", "3 (datacard)", "158" }

{ "", "Fastweb", "159" }

{ "", "Fastweb (datacard)", "160" }

{ "", "TIM", "50" }

{ "", "Vodafone", "51" }

{ "", "WIND", "52" }

MY (Malaysia)

{ "", "DIGI", "161" }

{ "", "Maxis", "162" }

{ "", "Timecel", "163" }

{ "", "TM Touch", "164" }

MX (Mexico)

{ "", "Telcel", "175" }

NL (Netherlands)

{ "", "Hi", "53" }

{ "", "KPN Mobile", "54" }

{ "", "O2", "55" }

{ "", "T-Mobile active", "56" }

{ "", "Telfort", "57" }

{ "", "Vodafone Live", "58" }

{ "", "Vodafone (business)", "59" }

NZ (New Zealand)

{ "", "Vodafone", "60"}

NO (Norway)

{ "", "Netcom", "63" }

{ "", "TalkMore", "64" }

{ "", "Telenor Mobil", "65" }

{ "", "Ventelo", "66" }

PE (Peru)

{ "", "Claro (TIM)", "67" }

PH (Philippines)

{ "", "Globe", "165" }

{ "", "Smart ", "166" }

PL (Poland)

{ "", "Orange", "124" }

{ "", "Polkomtel", "125" }

{ "", "ERA GSM", "126" }

{ "", "Play", "127" }

PT (Portugal)

{ "", "Zon", "128" }

{ "", "Vodafone Telecel", "129" }

{ "", "Optimus", "130" }

{ "", "Oniway", "131" }

{ "", "TMN", "132" }

RU (Russia)

{"", "MTS", "121" }

{ "", "Megafon", "122" }

{ "", "Beeline", "123" }

SG (Singapore)

{ "", "M1", "68" }

{ "", "SingTel", "69" }

{ "", "Starhub", "70" }

SK (Slovakia)

{ "", "Telefonica O2", "167" }

{ "", "T-Mobile", "168" }

{ "", "T-Mobile Plus", "169" }

{ "", "Orange", "170" }

ZA (South Africa)

{ "", "8ta", "185" }

{ "", "Cell-C", "71" }

{ "", "MTN", "72" }

{ "", "Virgin Mobile", "73" }

{ "", "Vodacom", "74" }

SE (Sweden)

{ "", "Tele2", "75" }

{ "", "Telenor(GPRS)", "77" }

{ "", "Telenor(3G)", "78" }

{ "", "Telia", "79" }

{ "", "Tre (3G)", "80" }

CH (Switzerland)

{"", "Swisscom", "133" }

{"", "Sunrise (T@ke Away)", "135" }

TW (Taiwan)

{ "", "Chunghwa Telecom", "81" }

{ "", "Far EasTone", "82" }

{ "", "Hinet", "83" }

{ "", "KG Telecom", "84" }

{ "", "Taiwan Cellular", "85" }

{ "", "Asia Pacific Telecom Group (APTG)", "86" }

{ "", "TransAsia", "87" }

TH (Thailand)

{ "", "AIS", "171" }

{ "", "DTAC", "172" }

AE (United Arab Emirates: Dubai)

{ "", "Etisalat", "152" }

GB (United Kingdom)

{ "", "3", "88" }

{ "", "3 (broadband)", "89" }

{ "", "AirTel Vodafone", "90" }

{ "", "Jersey Telecom", "91" }

{ "", "O2", "92" }

{ "", "O2 (optimised)", "93" }

{ "", "O2 (pay)", "109" }

{ "", "O2 (prepaid)", "94" }

{ "", "Orange (Pay Monthly)", "95" }

{ "", "Orange (JustTalk)", "96" }

{ "", "T-Mobile", "97" }

{ "", "Tesco Mobile", "98" }

{ "", "Virgin Mobile", "99" }

{ "", "Vodafone (contract)", "100" }

{ "", "Vodafone (pre-pay)", "101" }

US (United State of America)

{ "", "T-Mobile", "103" }

{ "", "AT&T", "104" }

Index

E

F

G

H

I

J

L

M

R

S

T

U

V

W

X

Z