Introduction

Cisco Unified Customer Voice Portal (Unified CVP) software has been designed to be easy to use but highly extendable. While the software provides enough to produce high quality voice applications out of the box, many users will want to extend the functionality of the software by building custom components that perform very specific tasks. This document describes the processes and application programming interfaces (APIs) provided for a developer to construct and deploy these components.

The components a developer uses the APIs to construct are: configurable action, decision, and voice elements, standard action and decision elements, dynamic element configurations, start and end of call actions, start and end of application actions, the on error notification, hotevents, Say It Smart plugins, and global and application loggers.

• Requirements, on page 1

Requirements

All components require programming effort to construct. In order to build these components, Unified CVP provides a Java API as well as an API that allows the use of other programming languages. Therefore, the reader should at least possess a familiarity with programming concepts.

Some components can only be constructed using the Java API and so for these components, the reader should possess a familiarity with the Java programming language. The reader should understand Java interfaces and abstract classes, extending classes and overriding methods, static variables and methods, Java collections, and compiling and deploying Java classes and JAR files. Unified CVP does not require very complex Java coding, knowing the basics of the Java language will be sufficient to start working with the Unified CVP Java API. Most of the information about the Java API is encapsulated in the API's Javadocs. This document will serve to provide a starting point and the Javadocs will provide the details.

Some components are used to produce VoiceXML, the language used to communicate with a voice browser. When building these components, a familiarity with VoiceXML is essential. The Java API used to produce VoiceXML follows the same design as the VoiceXML language, so a developer that understands how to write VoiceXML will be able to produce it using the API much easier and faster than a developer not familiar with VoiceXML.

Finally, the reader should familiarize themselves with the way Cisco Unified CVP VXML Server (VXML Server) works by reading the User Guide for Cisco Unified CVP VXML Server and Unified Call Studio, especially Chapter 2, which explains the purpose for each of the components described in this document. Familiarity with Cisco Unified Call Studio (Call Studio) and Builder for Call Studio is required for understanding how to construct configurable elements and Say It Smart plugins since they define how they are displayed within the Builder.

Note The grammar logic supplied with the out-of-the-box plug-in follows English grammar logic only. To achieve logic for other languages, you must develop your own plug-in.

Note You can create your custom elements or use additional Java classes in the Cisco Call Studio. If you need support in developing or troubleshooting it, you must have a developer support services contract or work with a Cisco partner/Cisco Advanced Services who has a developer support services contract.