

# **CAPLIN AUTH SDK 4.2**

## Overview

**June 2006**

<b>Preface</b>	<b>3</b>
What this document contains	3
Who should read this document	3
Typographical conventions	3
Acronyms and glossary	4
Feedback	5
 <b>Overview</b>	 <b>6</b>
What is an Auth Module?	6
The Caplin Platform architecture	7
How Auth Modules Work	9
Standard Auth Modules	10
Creating your own Auth Modules	11
Example Auth Module created using SDK	11
 <b>Openauth</b>	 <b>12</b>
 <b>Config Auth</b>	 <b>13</b>
 <b>XML Auth</b>	 <b>14</b>
 <b>Writing your own Java Auth module</b>	 <b>15</b>
Creating a Java Auth Module	15

---

# 1 Preface

## 1.1 What this document contains

This document gives an overview of the Auth Modules which come with Caplin Liberator and of how you can use the Auth Module SDK to create your own modules for controlling user authentication and object permissioning.

## 1.2 Who should read this document

This document is written for administrators wanting to get an idea of the different ready made auth modules which come with Caplin Liberator, and for developers needing to create custom Auth Modules for use with a Caplin Liberator. In the introduction to the Auth SDK this document describes functions and processes requiring a knowledge of programming in C and Java.

## 1.3 Typographical conventions

This document uses the following typographical conventions to identify particular elements within the text.

Type	Use
<b>Arial Bold</b>	Function names and methods. Other sections and chapters within this document.
<i>Arial Italic</i>	Parameter names and other variables.
<i>Times Italic</i>	File names, folders and directories.
Courier	Program output and code examples.
❖	Information bullet point.
■	Instruction.

---

## 1.4 Acronyms and glossary

<i>Auth Module</i>	An application that performs authentication and authorisation functions.
<i>Authentication</i>	Permitting a particular user to login to Caplin Liberator in order to access streaming data. Also known as entitlement.
<i>Authorisation</i>	Permitting particular data to be viewed.
<i>Caplin Liberator</i>	A suite of software applications and components for publishing real-time information using RTTP protocol over IP networks. Caplin Liberator collects real-time data from one or more sources and redistributes it to suitably permissioned RTTP subscribers.
<i>DACS</i>	Reuters's system for administering market data and permissioning trading floor, enterprise and Internet users to data, transactions, and other services.
<i>DTD</i>	Document Type Definition, a specification that accompanies an XML document and identifies what the tags are that separate paragraphs and how each is to be processed.
<i>Field</i>	A data element of an object, identified by a field name or field number and with a data value of a string.
<i>Object</i>	There are several types of RTTP object: Directory, Page, Record, News headline, News story and Chat object. Each type is identified by a three digit number.
<i>Parameter</i>	A data element of an object, identified by a field name or field number and with a data value of a string. Same as "field".
<i>Parser</i>	A program that receives input in the form of program instructions, markup tags or some other interface and breaks them up into parts (for example, objects, methods and their attributes) that can then be managed by other programs.

---

<i>RTTP</i>	RTTP (Real Time Text Protocol) is a web protocol developed by Caplin Systems Ltd that implements advanced real-time streaming for almost all types of textual information, including logical records, news and free-format pages.
<i>Symbols</i>	The letters used to uniquely identify a financial instrument (e.g. the symbol for Microsoft's common stock on the Nasdaq market is MSFT). Many symbol naming conventions (symbolologies) exist, but each is applied consistently in each market, and one symbology is used across all North American equity markets.
<i>UDP</i>	UDP (User Datagram Protocol) is a communications protocol that offers a limited amount of service when messages are exchanged between computers in a network that uses the Internet Protocol.
<i>XML</i>	XML (Extensible Markup Language) contains markup symbols to describe the contents of a page or file. An XML file can be processed purely as data by a program or it can be stored with similar data on another computer or displayed. XML is "extensible" because, unlike HTML, the markup symbols are unlimited and self-defining.
<i>XML tags</i>	The sequence of characters or other symbols that you insert at certain places in a file to indicate how the file should look when it is printed or displayed or to describe the document's logical structure. Tags that have other tags within them are called parent tags; those within are called child tags.

## 1.5 Feedback

Customer feedback can only improve the quality of Caplin product documentation, and we would welcome any comments, criticisms or suggestions you may have regarding this document.

Please email your thoughts to [documentation@caplin.com](mailto:documentation@caplin.com).

---

## 2 Overview

### 2.1 What is an Auth Module?

Caplin Liberator supports a modular system for handling authentication of users and entitlement of objects. This allows users to be authenticated, objects to have permissions loaded, read and write permissions for a user to be checked and object name mappings to be performed.

- ❖ Authentication is the process of determining whether someone is who they say they are. In networks such as the Internet, authentication is commonly done through the use of logon passwords: knowledge of the password is assumed to guarantee that the user is authentic. The user must know and use the declared password.
- ❖ Authorisation is the process of giving someone permission to do or have something. A system administrator defines which users are allowed access to which files. Authorisation is sometimes seen as both the preliminary setting up of permissions by a system administrator and the actual checking of the permission values that have been set up when a user is getting access.

For details on how to create your own Auth Modules, refer to the companion Javadoc and Doxygen documents **Caplin Liberator Auth Module SDK Developer's Guide** for Java or C respectively.

## 2.2 The Caplin Platform architecture

Figure 2-1 below shows a detailed illustration of the Caplin Platform architecture, including all available products and the protocols they use to communicate.

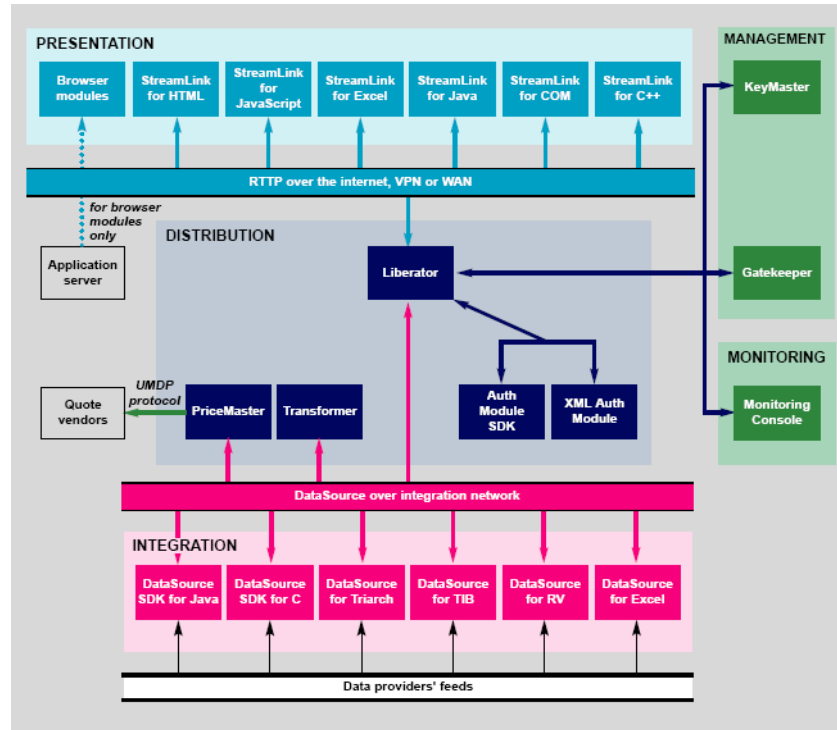


Figure 2-1: Caplin's architecture

Figure 2-2 below shows a simplified Caplin platform diagram and highlights an Auth Module and its place in the platform.

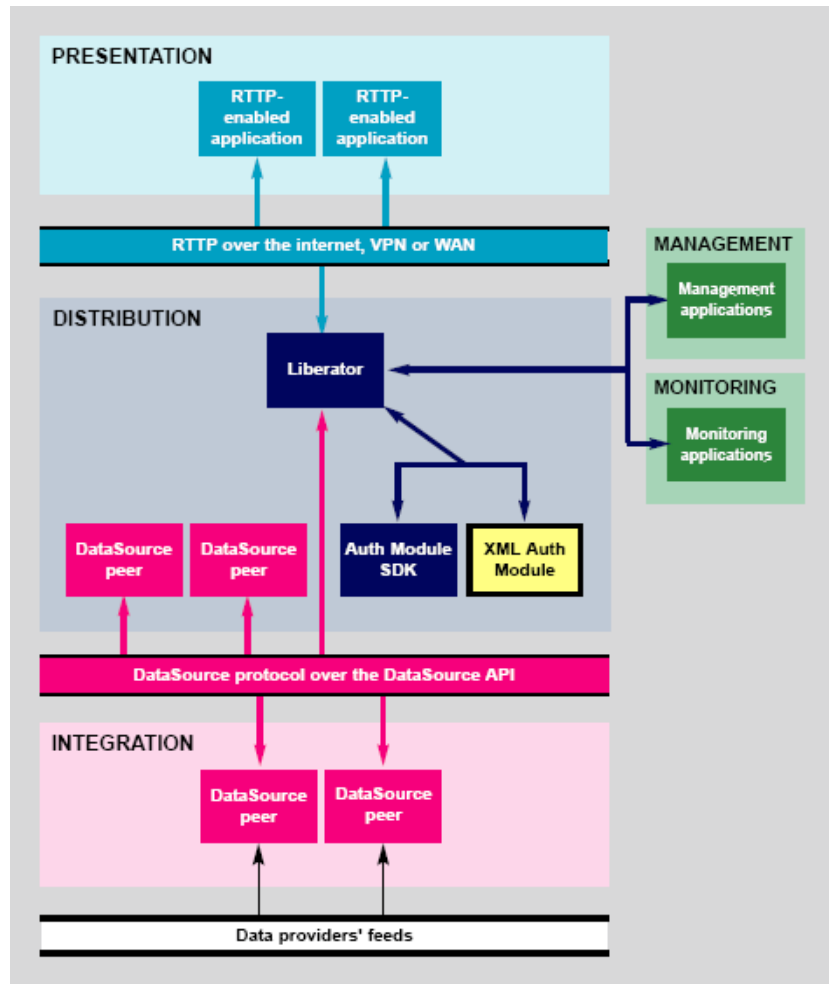
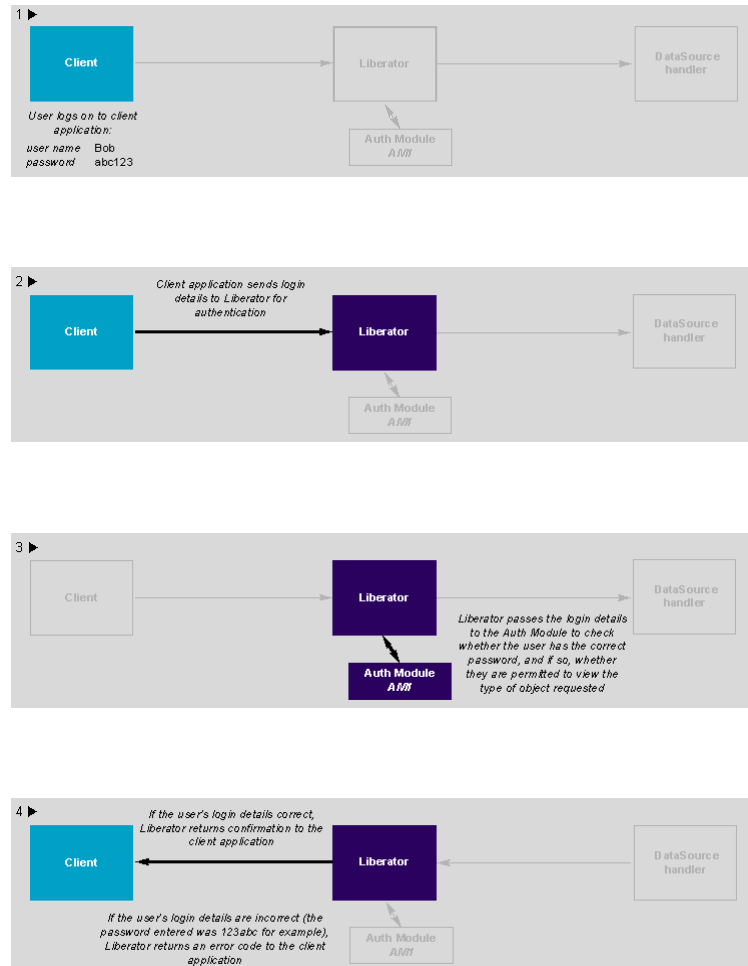


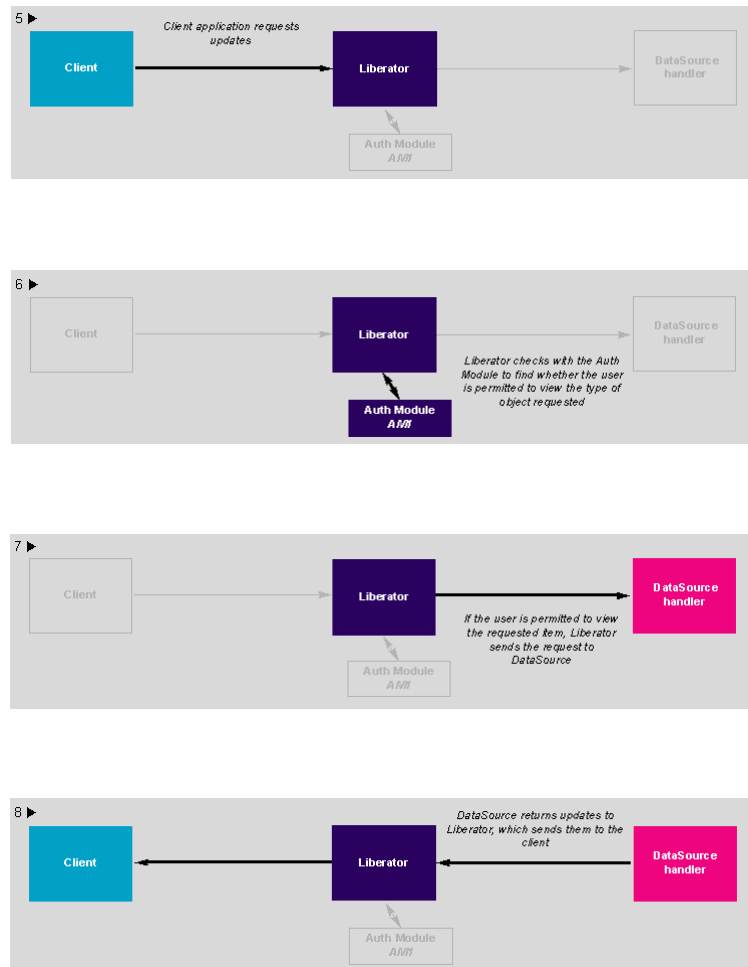
Figure 2-2: Auth Module in Caplin's architecture



## 2.3 How Auth Modules Work

The diagrams below give a step by step illustration of how auth modules are used to perform permissioning and entitlement.





## 2.4 Standard Auth Modules

Caplin Liberator is equipped with three standard Auth Modules: xmlauth, openauth and cfgauth.

---

These modules and their implementation are described in the companion document **Caplin Liberator Administration Guide**, in the chapter **Authentication and Entitlement** and also in the **XMLAuth Administration Guide**.

## 2.5 Creating your own Auth Modules

You can create your own auth modules using the Caplin Liberator Auth Module SDK.

## 2.6 Example Auth Module created using SDK

You can find an Auth Module, called `demoauth`, which is a basic implementation of an Auth Module that uses the functions and definitions described in this section in the `doc/examples` directory of the Caplin Liberator installation.

**Note:** *Most of the possible features are included, but many are contained in preprocessor blocks so by default they are not compiled.*

---

## 3 Openauth

This is the simplest Auth Module possible and can be used for systems where no authentication or authorisation is needed.

Openauth will allow any username to enter the system and with any password. It can also specify whether all users have either or both read and write access to any object in the system.

For further details please refer to the **Authentication and Entitlement** section of the **Liberator 4.2 Administration Guide**.

---

## 4 Config Auth

This module allows the number of users and the types of objects they can read to be configured.

This module is intended for relatively low numbers of users where the usernames and other details do not need to be changed often.

For further details please refer to the **Authentication and Entitlement** section of the **Liberator 4.2 Administration Guide**.

---

## 5 XML Auth

This module enables programmers and system administrators to use XML to create their own permissioning structures and control entitlement to objects held on Caplin Liberator.

As XMLauth is more complex than the other standard modules, there is an accompanying document **XML Auth 4.2 Admin Guide** which must be referred to for instructions on how to use this module.

## 6 Writing your own Java Auth module

The Caplin Liberator Java Authentication SDK is a library of classes that enables developers to create authentication modules for the Caplin Liberator. Caplin Liberator supports a modular system for handling authentication of users and entitlement of objects. This allows users to be authenticated for login, object-level read and write permissions for a user to be checked and object name mappings to be performed. Session management is also possible - actions such as ejection and re-validation of Liberator sessions are supported.

The primary functions of an Authentication module is to check the user on initial log-on and to check whether that user has permission to read from or write to a given object.

The Auth SDK allows you to create a variety of functions necessary for building an authentication system for Caplin Liberator, including functions to:

- ❖ Integrate with KeyMaster.
- ❖ Reject or invalidate users - for instance to reject a user when they try to log in with the same details as a user who is already logged on.
- ❖ Allow object-based permissioning - ie only allowing access to an object only if it contains a certain value in one of its fields, as opposed to being based on the symbol name of the object itself.
- ❖ Check the permissioning for each update.
- ❖ Return a range of codes which have no specified meaning. This allows an Auth Module and a custom client application to provide custom messages for denying access.
- ❖ To map a user-requested object name to a different name in the server (and therefore at upstream DataSource(s) too). This can be useful for providing different data under the same symbol name to different users or groups of users, for example in order to provide preferential currency spreads to certain customers, or customised data for particular logins.

### 6.1 Creating a Java Auth Module

An Auth module is created by implementing the Authenticator interface. There is a default implementation adaptor (AuthenticatorAdaptor that can be extended if only a few custom methods are needed. The API is callback-based. Calls are made on the various Authenticator

---

methods as and when an Auth action occurs in the Liberator. All the callback methods need to return an `AuthenticationResult` instance to represent the result of the Auth call.

There are two main features an Auth Module must implement:

❖ The initialisation function

This is an entry point into the module from the core server code.

See the companion Javadoc and Doxygen documents **Caplin Liberator Auth Module SDK Developer's Guide** for Java or C respectively for information on how to use this function.

❖ The function table

When Caplin Liberator loads an Auth Module, a pointer to a structure of function pointers is passed to the initialisation method. This structure must be filled with module-specific implementations of each function.

See the companion Javadoc and Doxygen documents **Caplin Liberator Auth Module SDK Developer's Guide** for Java or C respectively for information on how to implement this table.

This is all that is needed to set up and initialise the module. You may need to do some of your own initialisation of resources within the initialisation function. From this point it is left to the functions stored in the function table to handle their various tasks.





*The information contained in this publication is subject to UK, US and international copyright laws and treaties and all rights are reserved. No part of this publication may be reproduced or transmitted in any form or by any means without the written authorisation of an Officer of Caplin Systems Limited.*

*Various Caplin technologies described in this document are the subject of patent applications. All trademarks, company names, logos and service marks/names ("Marks") displayed in this publication are the property of Caplin or other third parties and may be registered trademarks. You are not permitted to use any Mark without the prior written consent of Caplin or the owner of that Mark.*

*This publication is provided "as is" without warranty of any kind, either express or implied, including, but not limited to, warranties of merchantability, fitness for a particular purpose, or non-infringement.*

*This publication could include technical inaccuracies or typographical errors and is subject to change without notice. Changes are periodically added to the information herein; these changes will be incorporated in new editions of this publication. Caplin Systems Limited may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.*

## Contact Us

Triton Court  
14 Finsbury Square  
London EC2A 1BR  
UK

Telephone: +44 20 7826 9600

Fax: +44 20 7826 9610

**[www.caplin.com](http://www.caplin.com)**

**[info@caplin.com](mailto:info@caplin.com)**