



## VIVOTEK NETWORK DEVELOPMENT PLATFORM

### **VNDP-MainProfile**

**Version 3.2**

**2017/01/20**

© 2017 VIVOTEK Inc. All Right Reserved

*VIVOTEK may make changes to specifications and product descriptions at any time, without notice.*

*The following is trademarks of VIVOTEK Inc., and may be used to identify VIVOTEK products only: VIVOTEK. Other product and company names contained herein may be trademarks of their respective owners.*

*No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from VIVOTEK Inc.*

## TABLE OF CONTENTS

---

TABLE OF CONTENTS .....	1
1. OVERVIEW.....	2
1.1 Introduction .....	2
1.2 Why Use MainProfile .....	2
2. GETTING STARTED .....	3
2.1 Architecture Overview for MainProfile .....	3
2.2 Sample Code.....	4
2.3 MainProfile Essentials .....	4
3. REVISION HISTORY .....	6

# 1. Overview

---

## 1.1 Introduction

---

VNDP MainProfile is a DLL-based developing tool supported by VIVOTEK to his Application Development Partners (ADPs) to help them quickly develop application that can use the maximum power of VIVOTEK Visual Server series. In this document, we will explain the concept of MainProfile and the relationship among the various modules in MainProfile. For details of each module, please refer to the document of individual module.

## 1.2 Why Use MainProfile

---

MainProfile hides many complexities of interaction with VIVOTEK's servers and IP cameras. It tries to unify the programming interfaces which are used to communicate with servers and IP cameras, hence it can minimize the ADPs' efforts when writing their applications in integration.

For developers that need to focus on the knowledge domain of their business fields, MainProfile can provide a very efficient implementation of live video/audio data collection and presentation. MainProfile tries to utilize the maximum hardware capabilities on the client systems so it could save you a lot of time to tune the performance.

## 2. Getting Started

---

### 2.1 Architecture Overview for MainProfile

---

MainProfile contains seven different modules which provide the following functions:

#### DataBroker

---

Handles the media data connection with servers and IP cameras. It carries the media data through callback functions during streaming. Users don't have to handle network connection handshaking and the parsing and reassembling of media frame. It also provides a tunnel for HTTP-only media (video only) to use the ability for frame reassemble capability. In this case, users need to handle the network connection on their own.

#### AVSynchronizer

---

Handles the decoding for encoded frames (either live media stream from Visual Server or stored media in storage) and the synchronization of audio and video data. Since this module uses the DirectX as the underlying video output mechanism, it could take advantage of the hardware acceleration ability in the system hence achieve a great performance. And this module also provides another channel which bypass the output of media and give users the decoded media frames. In this case, users could use the decoded frames to do their own business.

#### ServerManager

---

Provides a wrapper for the access of some common URL commands. Such as DI/DO, Uart (COM port on servers) read/write, config.ini retrieving and updating(for 2K, 3K series), set parameters and get parameters (for 6K, 7K, 8K series) and PTZ camera control. And if these functions can't fit your requirements, this module also provides a function which let you do HTTP POST or GET, so that you could send to server whatever command you need to talk to it.

#### DataPacketParser

---

Parses the raw data frame and converts it to a readable structure. In this structure, it contains the useful information about the media frame. Since the input interface of AVSynchronizer needs structure rather than raw frame data, you need to use this module to parse the raw data frame while using AVSynchronizer.

#### PacketMaker

---

Provides the capability to generate a formatted media packet from user's encoded data. It helps users to develop their own proxy server by converting MPEG4 to MJPEG packets. It also can generate client side audio packets so that the packet could be used in two-way communication.

## DRMControl

---

Provides developers a method to search servers or IP cameras which are alive in LAN. User can use this module to discover the servers or IP cameras which are equipped with DRM enabled firmware.

## AVComposer

---

Applications can use AVComposer to save media packets to files which are compatible for ISO base file format.

## AVReader

---

Applications can use AVReader to read media packets from files which are generated by AVComposer or grabbed by server's web plugin.

## 2.2 Sample Code

---

In the SDK package, we provide many samples to introduce you the usage of MainProfile, please refer to the package for the samples.

## 2.3 MainProfile Essentials

---

### HANDLE-Based API Design

---

Most of MainProfile modules require user to provide HANDLE in their API calls. A module's HANDLE is unique and can be viewed as a context which is pass to APIs at run time. Before using any module's API, you need to use the module's Initial (or Create) function to acquire resources in HANDLE. After that, when you doesn't need the module anymore, you should use the module's Release (or Delete) function to free the resources in HANDLE.

Do not mix up module's HANDLE at run time! If you use a HANDLE which is initiated by another Initial (or Create) function, it may cause application to work abnormally even crash.

### Version Compatibility

---

MainProfile uses version number to track the module release state and it represents the level of change in module, e.g. bug-fixes, add new API, ...etc. If you use the DLL with inappropriate version, it may cause compiler to complain or application to work abnormally.

Version number consists of four fields, which are major, minor, build and revision. Raise major version number means that some of existing module's interfaces (API, struct, ...etc) are changed. Raise minor version

number means that one or more new interfaces are add to module. Raise build version number often means that the modules performance is improved or its sub-module is upgraded. Raise revision version number means that one or several bug-fixes in module.

In short, raise of major version means that the new DLL is not backward-compatible to current product, so you need to re-compile product using new header files and import libraries. On the other hand, raise of other version number means that you can directly take the new DLL to run under current product.

### Callback-Based Flow

---

MainProfile uses the callback mechanism to pass information from inside of module to user. Take DataBroker as an example, while it creates internal threads to handle media streaming in network, it then uses callback functions to deliver current status or to carry media data.

You can implement the callback function according to the function type which is defined in module header file. After that, you need to set (or register) the pointer of callback function to module through API. Then modules will callback essential information at run time.

When you use callback function in your product, following usages are NOT recommended:

1. Put time-consuming work in callback function. Because the callback function is usually called by module's internal thread, blocking the callback function with long time may cause module to work abnormally.
2. Release module resources in callback function. Because module proceeds to handle the rest jobs after the callback function returns, if some resources are destroyed in callback function, it may cause module to work abnormally.

### 3. Revision History

---

Revision	Date	Description
3.2	2017/01/20	<ul style="list-style-type: none"><li>• Add AVComposer and AVReader modules.</li><li>• Deprecate ServerChannel module.</li><li>• Varius bug fixes.</li></ul>