**New SDK API**

No.：XX-XX-XX-XX

Date created：2021-07-03

Version No.：0.0.0.50

Logs Changed

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date Changed** | **User** | **Description** | **Version** |  |
| 2018-01-19 | HU | Create | 0.0.0.1 |  |
| 2018-03-28 | Dai | Add client connunication mode and 2.24--2.38 I/O | 0.0.0.14 |  |
| 2018-04-03 | Dai | Improve 2.37 ,2.38I/O 2.35change data format | 0.0.0.15 |  |
| 2018-05-18 | Dai | Add Face device I/O, | 0.0.0.16 |  |
| 2018-05-25 | Dai | Add 2.39--2.58 I/O | 0.0.0.17 |  |
| 2018-06-07 | Dai | Add 2.59--2.60 I/O | 0.0.0.18 |  |
| 2018-07-09 | Dai | Add customize function and improve demo | 0.0.0.19 |  |
| 2018-07-23 | Dai | Add 2.61--2.77 I/O | 0.0.0.21 |  |
| 2018-08-23 | Dai | Increase The Customize function | 0.0.0.23 |  |
| 2018-09-20 | Dai | Update the function for automatic add setup port of the server | 0.0.0.25 |  |
| 2018-11-29 | Dai | Update the user ID up to 8 digit | 0.0.0.26 |  |
| 2019-01-18 | Dai | Add the Log manage | 0.0.0.28 |  |
| 2021-07-02 | Dai | Support Facedeep series | 0.0.0.50 |  |

**Content**

[1 Overview 24](#_Toc76388941)

[1.1 Introduction 24](#_Toc76388942)

[1.2 Overview of functions 24](#_Toc76388943)

[1.3 System logic structure 24](#_Toc76388944)

[1.4 Interface functions 25](#_Toc76388945)

[1.4.2 Face Deep Series Temperature detection and template 28](#_Toc76388946)

[1.5 Message Type 29](#_Toc76388947)

[1.6 Setup label and print setup Description 34](#_Toc76388948)

[2 Description interface 36](#_Toc76388949)

[2.1 CChex\_Version 36](#_Toc76388950)

[2.1.1 Description functions 36](#_Toc76388951)

[2.1.2 Request 36](#_Toc76388952)

[2.1.3 Response 36](#_Toc76388953)

[2.1.4 Sample 36](#_Toc76388954)

[2.1.5 Notice 36](#_Toc76388955)

[2.2 CChex\_Init 36](#_Toc76388956)

[2.2.1 Description functions 36](#_Toc76388957)

[2.2.2 Request 36](#_Toc76388958)

[2.2.3 Response 37](#_Toc76388959)

[2.2.4 Sample 37](#_Toc76388960)

[2.2.5 Notice 37](#_Toc76388961)

[2.3 CChex\_Start 37](#_Toc76388962)

[2.3.1 Description functions 37](#_Toc76388963)

[2.3.2 Request 37](#_Toc76388964)

[2.3.3 Response 37](#_Toc76388965)

[2.3.4 Sample 37](#_Toc76388966)

[2.3.5 Notice 38](#_Toc76388967)

[2.4 CChex\_Stop 38](#_Toc76388968)

[2.4.1 Description functions 38](#_Toc76388969)

[2.4.2 Request 38](#_Toc76388970)

[2.4.3 Response 38](#_Toc76388971)

[2.4.4 Sample 38](#_Toc76388972)

[2.4.5 Notice 38](#_Toc76388973)

[2.5 CChex\_Update 39](#_Toc76388974)

[2.5.1 Description functions 39](#_Toc76388975)

[2.5.2 Request 39](#_Toc76388976)

[2.5.3 Response 39](#_Toc76388977)

[2.5.4 Sample 39](#_Toc76388978)

[2.5.5 Notice 40](#_Toc76388979)

[2.6 CChex\_GetNetConfig 40](#_Toc76388980)

[2.6.1 Description functions 40](#_Toc76388981)

[2.6.2 Request 40](#_Toc76388982)

[2.6.3 Response 41](#_Toc76388983)

[2.6.4 Sample 41](#_Toc76388984)

[2.6.5 Notice 41](#_Toc76388985)

[2.7 CChex\_SetNetConfig 42](#_Toc76388986)

[2.7.1 Description functions 42](#_Toc76388987)

[2.7.2 Request 42](#_Toc76388988)

[2.7.3 Response 42](#_Toc76388989)

[2.7.4 Sample 43](#_Toc76388990)

[2.7.5 Notice 43](#_Toc76388991)

[2.8 CChex\_MsgGetByIdx 43](#_Toc76388992)

[2.8.1 Description functions 43](#_Toc76388993)

[2.8.2 Request 43](#_Toc76388994)

[2.8.3 Response 43](#_Toc76388995)

[2.8.4 Sample 44](#_Toc76388996)

[2.8.5 Notice 44](#_Toc76388997)

[2.9 CChex\_MsgDelByIdx 44](#_Toc76388998)

[2.9.1 Description functions 44](#_Toc76388999)

[2.9.2 Request 45](#_Toc76389000)

[2.9.3 Response 45](#_Toc76389001)

[2.9.4 Sample 45](#_Toc76389002)

[2.9.5 Notice 45](#_Toc76389003)

[2.10 CChex\_MsgAddNew 45](#_Toc76389004)

[2.10.1 Description functions 45](#_Toc76389005)

[2.10.2 Request 46](#_Toc76389006)

[2.10.3 Response 46](#_Toc76389007)

[2.10.4 Sample 47](#_Toc76389008)

[2.10.5 Notice 47](#_Toc76389009)

[2.11 CChex\_MsgGetAllHead 47](#_Toc76389010)

[2.11.1 Description functions 47](#_Toc76389011)

[2.11.2 Request 47](#_Toc76389012)

[2.11.3 Response 47](#_Toc76389013)

[2.11.4 Sample 48](#_Toc76389014)

[2.11.5 Notice 48](#_Toc76389015)

[2.12 CChex\_RebootDevice 48](#_Toc76389016)

[2.12.1 Description functions 48](#_Toc76389017)

[2.12.2 Request 48](#_Toc76389018)

[2.12.3 Response 49](#_Toc76389019)

[2.12.4 Sample 49](#_Toc76389020)

[2.12.5 Notice 49](#_Toc76389021)

[2.13 CChex\_SetTime 49](#_Toc76389022)

[2.13.1 Description functions 49](#_Toc76389023)

[2.13.2 Request 49](#_Toc76389024)

[2.13.3 Response 50](#_Toc76389025)

[2.13.4 Sample 50](#_Toc76389026)

[2.13.5 Notice 50](#_Toc76389027)

[2.14 CChex\_GetSNConfig 50](#_Toc76389028)

[2.14.1 Description functions 50](#_Toc76389029)

[2.14.2 Request 50](#_Toc76389030)

[2.14.3 Response 50](#_Toc76389031)

[2.14.4 Sample 51](#_Toc76389032)

[2.14.5 Notice 51](#_Toc76389033)

[2.15 CChex\_DownloadAllRecords 51](#_Toc76389034)

[2.15.1 Description functions 51](#_Toc76389035)

[2.15.2 Request 51](#_Toc76389036)

[2.15.3 Response 51](#_Toc76389037)

[2.15.4 Sample 52](#_Toc76389038)

[2.15.5 Notice 52](#_Toc76389039)

[2.16 CChex\_DeleteRecordInfo 52](#_Toc76389040)

[2.16.1 Description functions 52](#_Toc76389041)

[2.16.2 Request 52](#_Toc76389042)

[2.16.3 Response 53](#_Toc76389043)

[2.16.4 Sample 53](#_Toc76389044)

[2.16.5 Notice 53](#_Toc76389045)

[2.17 CChex\_GetBasicConfigInfo 53](#_Toc76389046)

[2.17.1 Description functions 54](#_Toc76389047)

[2.17.2 Request 54](#_Toc76389048)

[2.17.3 Response 54](#_Toc76389049)

[2.17.4 Sample 55](#_Toc76389050)

[2.17.5 Notice 55](#_Toc76389051)

[2.18 CChex\_SetBasicConfigInfo 55](#_Toc76389052)

[2.18.1 Description functions 55](#_Toc76389053)

[2.18.2 Request 55](#_Toc76389054)

[2.18.3 Response 56](#_Toc76389055)

[2.18.4 Sample 56](#_Toc76389056)

[2.18.5 Notice 56](#_Toc76389057)

[2.19 CChex\_ListPersonInfo 56](#_Toc76389058)

[2.19.1 Description functions 56](#_Toc76389059)

[2.19.2 Request 57](#_Toc76389060)

[2.19.3 Response 57](#_Toc76389061)

[2.19.4 Sample 58](#_Toc76389062)

[2.19.5 Notice 58](#_Toc76389063)

[2.20 CChex\_ModifyPersonInfo 58](#_Toc76389064)

[2.20.1 Description functions 58](#_Toc76389065)

[2.20.2 Request 58](#_Toc76389066)

[2.20.3 Response 60](#_Toc76389067)

[2.20.4 Sample 60](#_Toc76389068)

[2.20.5 Notice 60](#_Toc76389069)

[2.21 CChex\_DeletePersonInfo 60](#_Toc76389070)

[2.21.1 Description functions 60](#_Toc76389071)

[2.21.2 Request 60](#_Toc76389072)

[2.21.3 Response 61](#_Toc76389073)

[2.21.4 Sample 61](#_Toc76389074)

[2.21.5 Notice 61](#_Toc76389075)

[2.22 CChex\_DeletePersonInfo\_VER\_4\_NEWID 61](#_Toc76389076)

[2.22.1 Description functions 62](#_Toc76389077)

[2.22.2 Request 62](#_Toc76389078)

[2.22.3 Response 62](#_Toc76389079)

[2.22.4 Sample 63](#_Toc76389080)

[2.22.5 Notice 63](#_Toc76389081)

[2.23 CChex\_DownloadFingerPrint 63](#_Toc76389082)

[2.23.1 Description functions 63](#_Toc76389083)

[2.23.2 Request 63](#_Toc76389084)

[2.23.3 Response 63](#_Toc76389085)

[2.23.4 Sample 64](#_Toc76389086)

[2.23.5 Notice 64](#_Toc76389087)

[2.24 CChex\_DownloadFingerPrint\_VER\_4\_NEWID 64](#_Toc76389088)

[2.24.1 Function 64](#_Toc76389089)

[2.24.2 Request 64](#_Toc76389090)

[2.24.3 Respond 65](#_Toc76389091)

[2.24.4 Sample 65](#_Toc76389092)

[2.24.5 Notice 65](#_Toc76389093)

[2.25 CChex\_UploadFingerPrint 65](#_Toc76389094)

[2.25.1 Description functions 65](#_Toc76389095)

[2.25.2 Request 65](#_Toc76389096)

[2.25.3 Response 66](#_Toc76389097)

[2.25.4 Sample 66](#_Toc76389098)

[2.25.5 Notice 66](#_Toc76389099)

[2.26 CChex\_UploadFingerPrint\_VER\_4\_NEWID 66](#_Toc76389100)

[2.26.1 Function 67](#_Toc76389101)

[2.26.2 Request 67](#_Toc76389102)

[2.26.3 Respond 67](#_Toc76389103)

[2.26.4 Sample 68](#_Toc76389104)

[2.26.5 Notice 68](#_Toc76389105)

[2.27 CChex\_GetTime 68](#_Toc76389106)

[2.27.1 Description functions 68](#_Toc76389107)

[2.27.2 Request 68](#_Toc76389108)

[2.27.3 Response 68](#_Toc76389109)

[2.27.4 Sample 69](#_Toc76389110)

[2.27.5 Notice 69](#_Toc76389111)

[2.28 CChex\_InitUserArea 69](#_Toc76389112)

[2.28.1 Description functions 69](#_Toc76389113)

[2.28.2 Request 69](#_Toc76389114)

[2.28.3 Response 70](#_Toc76389115)

[2.28.4 Sample 70](#_Toc76389116)

[2.28.5 Notice 70](#_Toc76389117)

[2.29 CChex\_InitSystem 70](#_Toc76389118)

[2.29.1 Description functions 70](#_Toc76389119)

[2.29.2 Request 70](#_Toc76389120)

[2.29.3 Response 71](#_Toc76389121)

[2.29.4 Sample 71](#_Toc76389122)

[2.29.5 Notice 71](#_Toc76389123)

[2.30 CChex\_GetBasicConfigInfo2 71](#_Toc76389124)

[2.30.1 Description functions 71](#_Toc76389125)

[2.30.2 Request 72](#_Toc76389126)

[2.30.3 Response 72](#_Toc76389127)

[2.30.4 Sample 73](#_Toc76389128)

[2.30.5 Notice 73](#_Toc76389129)

[2.31 CChex\_SetBasicConfigInfo2 73](#_Toc76389130)

[2.31.1 Description functions 73](#_Toc76389131)

[2.31.2 Request 73](#_Toc76389132)

[2.31.3 Response 74](#_Toc76389133)

[2.31.4 Sample 74](#_Toc76389134)

[2.31.5 Notice 74](#_Toc76389135)

[2.32 CChex\_GetPeriodTime 75](#_Toc76389136)

[2.32.1 Description functions 75](#_Toc76389137)

[2.32.2 Request 75](#_Toc76389138)

[2.32.3 Response 75](#_Toc76389139)

[2.32.4 Sample 76](#_Toc76389140)

[2.32.5 Notice 76](#_Toc76389141)

[2.33 CChex\_SetPeriodTime 76](#_Toc76389142)

[2.33.1 Description functions 76](#_Toc76389143)

[2.33.2 Request 76](#_Toc76389144)

[2.33.3 Response 77](#_Toc76389145)

[2.33.4 Sample 77](#_Toc76389146)

[2.33.5 Notice 77](#_Toc76389147)

[2.34 CChex\_GetTeamInfo 77](#_Toc76389148)

[2.34.1 Description functions 77](#_Toc76389149)

[2.34.2 Request 78](#_Toc76389150)

[2.34.3 Response 78](#_Toc76389151)

[2.34.4 Sample 78](#_Toc76389152)

[2.34.5 Notice 78](#_Toc76389153)

[2.35 CChex\_SetTeamInfo 78](#_Toc76389154)

[2.35.1 Description functions 78](#_Toc76389155)

[2.35.2 Request 79](#_Toc76389156)

[2.35.3 Response 79](#_Toc76389157)

[2.35.4 Sample 79](#_Toc76389158)

[2.35.5 Notice 80](#_Toc76389159)

[2.36 CCHex\_AddFingerprintOnline 80](#_Toc76389160)

[2.36.1 Description functions 80](#_Toc76389161)

[2.36.2 Request 80](#_Toc76389162)

[2.36.3 Response 80](#_Toc76389163)

[2.36.4 Sample 81](#_Toc76389164)

[2.36.5 Notice 81](#_Toc76389165)

[2.37 CCHex\_ForcedUnlock 81](#_Toc76389166)

[2.37.1 Description functions 81](#_Toc76389167)

[2.37.2 Request 81](#_Toc76389168)

[2.37.3 Response 81](#_Toc76389169)

[2.37.4 Sample 82](#_Toc76389170)

[2.37.5 Notice 82](#_Toc76389171)

[2.38 CCHex\_Udp\_Search\_Dev 82](#_Toc76389172)

[2.38.1 Description functions 82](#_Toc76389173)

[2.38.2 Request 82](#_Toc76389174)

[2.38.3 Response 82](#_Toc76389175)

[2.38.4 Sample 85](#_Toc76389176)

[2.38.5 Notice 85](#_Toc76389177)

[2.39 CCHex\_Udp\_Set\_Dev\_Config 85](#_Toc76389178)

[2.39.1 Description functions 85](#_Toc76389179)

[2.39.2 Request 85](#_Toc76389180)

[2.39.3 Response 86](#_Toc76389181)

[2.39.4 Sample 86](#_Toc76389182)

[2.39.5 Notice 86](#_Toc76389183)

[2.40 CCHex\_ClientConnect 86](#_Toc76389184)

[2.40.1 Description functions 87](#_Toc76389185)

[2.40.2 Request 87](#_Toc76389186)

[2.40.3 Response 87](#_Toc76389187)

[2.40.4 Sample 88](#_Toc76389188)

[2.40.5 Notice 88](#_Toc76389189)

[2.41 CCHex\_ClientDisconnect 88](#_Toc76389190)

[2.41.1 Description functions 88](#_Toc76389191)

[2.41.2 Request 88](#_Toc76389192)

[2.41.3 Response 88](#_Toc76389193)

[2.41.4 Sample 89](#_Toc76389194)

[2.41.5 Notice 89](#_Toc76389195)

[2.42 CChex\_GetInfomationCode 89](#_Toc76389196)

[2.42.1 Description function 89](#_Toc76389197)

[2.42.2 Request 90](#_Toc76389198)

[2.40.3 Response 90](#_Toc76389199)

[2.42.4 Sample 90](#_Toc76389200)

[2.42.5 Notice 90](#_Toc76389201)

[2.43 CChex\_SetInfomationCode 90](#_Toc76389202)

[2.43.1 Description function 90](#_Toc76389203)

[2.43.2 Request 91](#_Toc76389204)

[2.43.3Response 91](#_Toc76389205)

[2.43.4 Sample 91](#_Toc76389206)

[2.43.5 Notice 91](#_Toc76389207)

[2.44 CChex\_GetBellInfo 92](#_Toc76389208)

[2.44.1 Description function 92](#_Toc76389209)

[2.44.2 Request 92](#_Toc76389210)

[2.44.3 Response 92](#_Toc76389211)

[2.44.4 Sample 93](#_Toc76389212)

[2.44.5 Notice 93](#_Toc76389213)

[2.45 CChex\_SetBellInfo 93](#_Toc76389214)

[2.45.1 Description function 93](#_Toc76389215)

[2.45.2 Request 93](#_Toc76389216)

[2.45.3 Response 94](#_Toc76389217)

[2.45.4 Sample 94](#_Toc76389218)

[2.45.5 Notice 94](#_Toc76389219)

[2.46 CChex\_GetUserAttendanceStatusInfo 94](#_Toc76389220)

[2.46.1 Description function 94](#_Toc76389221)

[2.46.2 Request 94](#_Toc76389222)

[2.46.3 Response 95](#_Toc76389223)

[2.46.4 Sample 95](#_Toc76389224)

[2.46.5 Notice 95](#_Toc76389225)

[2.47 CChex\_SetUserAttendanceStatusInfo 95](#_Toc76389226)

[2.47.1 Description function 95](#_Toc76389227)

[2.47.2 Request 96](#_Toc76389228)

[2.47.3 Response 96](#_Toc76389229)

[2.47.4 Sample 96](#_Toc76389230)

[2.47.5 Notice 96](#_Toc76389231)

[2.48 CChex\_ClearAdministratFlag 97](#_Toc76389232)

[2.48.1 Description function 97](#_Toc76389233)

[2.48.2 Request 97](#_Toc76389234)

[2.48.3 Response 97](#_Toc76389235)

[2.48.4 Sample 97](#_Toc76389236)

[2.48.5 Notice 97](#_Toc76389237)

[2.49 CChex\_GetSpecialStatus 98](#_Toc76389238)

[2.49.1 Description function 98](#_Toc76389239)

[2.49.2 Request 98](#_Toc76389240)

[2.49.3 Response 98](#_Toc76389241)

[2.49.4 Sample 99](#_Toc76389242)

[2.49.5 Notice 99](#_Toc76389243)

[2.50 CChex\_GetAdminCardnumberPassword 99](#_Toc76389244)

[2.50.1 Description function 99](#_Toc76389245)

[2.50.2 Request 99](#_Toc76389246)

[2.50.3 Response 99](#_Toc76389247)

[2.50.4 Sample 100](#_Toc76389248)

[2.50.5 Notice 100](#_Toc76389249)

[2.51 CChex\_SetAdminCardnumberPassword 100](#_Toc76389250)

[2.51.1 Description function 100](#_Toc76389251)

[2.51.2 Request 101](#_Toc76389252)

[2.51.3 Response 101](#_Toc76389253)

[2.51.4 Sample 101](#_Toc76389254)

[2.51.5 Notice 101](#_Toc76389255)

[2.52 CChex\_GetDSTParam 101](#_Toc76389256)

[2.52.1 Description function 101](#_Toc76389257)

[2.52.2Request 102](#_Toc76389258)

[2.52.3 Response 102](#_Toc76389259)

[2.52.4 Sample 103](#_Toc76389260)

[2.52.5 Notice 103](#_Toc76389261)

[2.53 CChex\_SetDSTParam 103](#_Toc76389262)

[2.53.1 Description function 103](#_Toc76389263)

[2.53.2 Request 103](#_Toc76389264)

[2.53.3 Response 104](#_Toc76389265)

[2.53.4 Sample 104](#_Toc76389266)

[2.53.5 Notice 104](#_Toc76389267)

[2.54 CChex\_GetDevExtInfo 104](#_Toc76389268)

[2.54.1 Description function 105](#_Toc76389269)

[2.54.2 Request 105](#_Toc76389270)

[2.54.3 Response 105](#_Toc76389271)

[2.54.4 Sample 105](#_Toc76389272)

[2.54.5 Notice 106](#_Toc76389273)

[2.55 CChex\_SetDevExtInfo 106](#_Toc76389274)

[2.55.1 Description function 106](#_Toc76389275)

[2.55.2 Request 106](#_Toc76389276)

[2.55.3 Response 106](#_Toc76389277)

[2.55.4 Sample 107](#_Toc76389278)

[2.55.5 Notice 107](#_Toc76389279)

[2.56 CChex\_GetBasicConfigInfo3 107](#_Toc76389280)

[2.56.1Description function 107](#_Toc76389281)

[2.56.2 Request 107](#_Toc76389282)

[2.56.3 Response 107](#_Toc76389283)

[2.56.4 Sample 108](#_Toc76389284)

[2.56.5 Notice 108](#_Toc76389285)

[2.57 CChex\_SetBasicConfigInfo3 108](#_Toc76389286)

[2.57.1 Description function 108](#_Toc76389287)

[2.57.2 Request 109](#_Toc76389288)

[2.57.3 Response 109](#_Toc76389289)

[2.57.4 Sample 110](#_Toc76389290)

[2.57.5 Notice 110](#_Toc76389291)

[2.58 CChex\_ConnectionAuthentication 110](#_Toc76389292)

[2.58.1 Description function 110](#_Toc76389293)

[2.58.2 Request 110](#_Toc76389294)

[2.58.3 Response 111](#_Toc76389295)

[2.58.4 Sample 111](#_Toc76389296)

[2.58.5 Notice 111](#_Toc76389297)

[2.59 CChex\_GetRecordNumByEmployeeIdAndTime 111](#_Toc76389298)

[2.59.1 Description function 111](#_Toc76389299)

[2.59.2 Request 111](#_Toc76389300)

[2.59.3 Response 112](#_Toc76389301)

[2.59.4 Sample 112](#_Toc76389302)

[2.59.5 Notice 112](#_Toc76389303)

[2.60 CChex\_DownloadRecordByEmployeeIdAndTime 113](#_Toc76389304)

[2.60.1 Description function 113](#_Toc76389305)

[2.60.2 Request 113](#_Toc76389306)

[2.60.3Response 113](#_Toc76389307)

[2.60.4 Sample 114](#_Toc76389308)

[2.60.5 Notice 114](#_Toc76389309)

[2.61 Send attendance records in real time 114](#_Toc76389310)

[2.61.1 Description function 114](#_Toc76389311)

[2.61.2 Request 114](#_Toc76389312)

[2.61.3 Response 115](#_Toc76389313)

[2.61.4Sample 115](#_Toc76389314)

[2.61.5 Notice 115](#_Toc76389315)

[2.62 CChex\_GetRecordInfo 115](#_Toc76389316)

[2.62.1 Description function 115](#_Toc76389317)

[2.62.2 Request 116](#_Toc76389318)

[2.62.3 Response 116](#_Toc76389319)

[2.62.4 Sample 116](#_Toc76389320)

[2.62.5 Notice 117](#_Toc76389321)

[2.63 CChex\_DownloadAllNewRecords 117](#_Toc76389322)

[2.63.1 Description function 117](#_Toc76389323)

[2.63.2Request 117](#_Toc76389324)

[2.63.3 Response 117](#_Toc76389325)

[2.63.4 Sample 118](#_Toc76389326)

[2.63.5 Notice 118](#_Toc76389327)

[2.64 Upload user information 118](#_Toc76389328)

[2.64.2 Request 118](#_Toc76389329)

[2.64.3 Response 121](#_Toc76389330)

[2.64.4 Sample 122](#_Toc76389331)

[2.64.5 Notice 122](#_Toc76389332)

[2.65 Chex\_GetBasicConfigInfo5 122](#_Toc76389333)

[2.65.1 Function 122](#_Toc76389334)

[2.65.2 Request 123](#_Toc76389335)

[2.65.3 Respond 123](#_Toc76389336)

[2.65.4 Sample 123](#_Toc76389337)

[2.65.5 Notice 123](#_Toc76389338)

[2.66 CChex\_SetBasicConfigInfo 5 123](#_Toc76389339)

[2.66.1 Function 123](#_Toc76389340)

[2.66.2 Request 124](#_Toc76389341)

[2.66.3 Respond 124](#_Toc76389342)

[2.66.4 Sample 124](#_Toc76389343)

[2.66.5 Notice 124](#_Toc76389344)

[2.67 CChex\_GetCardNo 125](#_Toc76389345)

[2.67.1 Function 125](#_Toc76389346)

[2.67.2 Request 125](#_Toc76389347)

[2.67.3 Respond 125](#_Toc76389348)

[2.67.4 Sample 125](#_Toc76389349)

[2.67.5 Notice 126](#_Toc76389350)

[2.68 CChex\_SetDevCurrentStatus 126](#_Toc76389351)

[2.68.1 Function 126](#_Toc76389352)

[2.68.2 Request 126](#_Toc76389353)

[2.68.3 Respond 126](#_Toc76389354)

[2.68.4 Sample 127](#_Toc76389355)

[2.68.5 Notice 127](#_Toc76389356)

[2.69 CChex\_GetServiceURL 127](#_Toc76389357)

[2.69.1 Function 127](#_Toc76389358)

[2.69.2 Request 127](#_Toc76389359)

[2.69.3 Response 127](#_Toc76389360)

[2.69.4 Sample 128](#_Toc76389361)

[2.69.5 Notice 128](#_Toc76389362)

[2.70 CChex\_SetServiceURL 128](#_Toc76389363)

[2.70.1 Function 128](#_Toc76389364)

[2.70.2 Request 128](#_Toc76389365)

[2.70.3 Response 129](#_Toc76389366)

[2.70.4 Sample 129](#_Toc76389367)

[2.70.15 Notice 129](#_Toc76389368)

[2.71 CChex\_UploadFile 129](#_Toc76389369)

[2.71.1 Function 129](#_Toc76389370)

[2.71.2 Request 129](#_Toc76389371)

[2.71.3 Response 130](#_Toc76389372)

[2.71.4 Sample 130](#_Toc76389373)

[2.71.5 Notice 130](#_Toc76389374)

[2.72 CChex\_UpdateDevStatus 131](#_Toc76389375)

[2.72.1 Function 131](#_Toc76389376)

[2.72.2 Request 131](#_Toc76389377)

[2.72.3 Response 131](#_Toc76389378)

[2.72.4 Sample 131](#_Toc76389379)

[2.72.5 Notice 132](#_Toc76389380)

[2.73 CChex\_GetStatusSwitch 132](#_Toc76389381)

[2.73.1 Function 132](#_Toc76389382)

[2.73.2 Request 132](#_Toc76389383)

[2.73.3 Response 132](#_Toc76389384)

[2.73.4 Sample 133](#_Toc76389385)

[2.73.5 Notice 133](#_Toc76389386)

[2.74 CChex\_SetStatusSwitch 133](#_Toc76389387)

[2.74.1 Function 133](#_Toc76389388)

[2.74.2 Request 133](#_Toc76389389)

[2.74.3 Response 134](#_Toc76389390)

[2.74.4 Sample 134](#_Toc76389391)

[2.74.5 Notice 135](#_Toc76389392)

[2.75 CChex\_GetStatusSwitch\_EXT 135](#_Toc76389393)

[2.75.1 Function 135](#_Toc76389394)

[2.75.2 Request 135](#_Toc76389395)

[2.75.3 Response 135](#_Toc76389396)

[2.75.4 Sample 136](#_Toc76389397)

[2.75.5 Notice 136](#_Toc76389398)

[2.76 CChex\_SetStatusSwitch\_EXT 136](#_Toc76389399)

[2.76.1 Function 136](#_Toc76389400)

[2.76.2 Request 136](#_Toc76389401)

[2.76.3 Response 137](#_Toc76389402)

[2.76.4 Sample 137](#_Toc76389403)

[2.76.5 Notice 137](#_Toc76389404)

[2.77 CChex\_Get\_Service\_Port 138](#_Toc76389405)

[2.77.1 Function 138](#_Toc76389406)

[2.77.2 Request 138](#_Toc76389407)

[2.77.3 Sample 138](#_Toc76389408)

[2.78 CChex\_SetSdkConfig 138](#_Toc76389409)

[2.78.1 Function 138](#_Toc76389410)

[2.78.2 Request 139](#_Toc76389411)

[2.78.3 Sample 139](#_Toc76389412)

[2.79 CChex\_UploadRecord 139](#_Toc76389413)

[2.79.1 Function 139](#_Toc76389414)

[2.79.2 Request 139](#_Toc76389415)

[2.79.3 Respond 140](#_Toc76389416)

[2.79.4 Sample 140](#_Toc76389417)

[2.79.5 Notice 140](#_Toc76389418)

[2.80 CChex\_UploadRecord\_VER\_4\_NEWID 140](#_Toc76389419)

[2.80.1 Function 140](#_Toc76389420)

[2.80.2 Request 141](#_Toc76389421)

[2.80.3 Respond 141](#_Toc76389422)

[2.80.4 Sample 141](#_Toc76389423)

[2.80.5 Notice 142](#_Toc76389424)

[2.81 CChex\_GetOnePersonInfo 142](#_Toc76389425)

[CChex\_GetOnePersonInfo\_VER\_4\_NEWID 142](#_Toc76389426)

[2.81.1 Function 142](#_Toc76389427)

[2.81.2 Request 142](#_Toc76389428)

[2.81.3 Respond 143](#_Toc76389429)

[2.81.4 Sample 146](#_Toc76389430)

[2.81.5 Notice 146](#_Toc76389431)

[2.82 CChex\_GetMachineId 147](#_Toc76389432)

[2.82.1 Function 147](#_Toc76389433)

[2.82.2 Request 147](#_Toc76389434)

[2.82.3 Respond 147](#_Toc76389435)

[2.82.4 Sample 147](#_Toc76389436)

[2.82.5 Notice 147](#_Toc76389437)

[2.83 CChex\_SetMachineId 148](#_Toc76389438)

[2.83.1 Function 148](#_Toc76389439)

[2.83.2 Request 148](#_Toc76389440)

[2.83.3 Respond 148](#_Toc76389441)

[2.83.4 Sample 148](#_Toc76389442)

[2.83.5 Notice 149](#_Toc76389443)

[2.84 CChex\_ManageLogRecord 149](#_Toc76389444)

[2.84.1 Function 149](#_Toc76389445)

[2.84.2 Request 149](#_Toc76389446)

[2.84.3 Respond 150](#_Toc76389447)

[2.84.4 Sample 151](#_Toc76389448)

[2.84.5 Notice 151](#_Toc76389449)

[3 Face Deep Series Temperature detection and face template interface 152](#_Toc76389450)

[3.1 CChex\_TM\_DownloadAllRecords 152](#_Toc76389451)

[3.1.1 Function Description 152](#_Toc76389452)

[3.1.2 Request 152](#_Toc76389453)

[3.1.3 Response Interface 152](#_Toc76389454)

[3.1.4 Sample 153](#_Toc76389455)

[3.1.5 Notice 153](#_Toc76389456)

[3.2 CChex\_TM\_DownloadAllNewRecords 153](#_Toc76389457)

[3.2.1 Function Description 153](#_Toc76389458)

[3.2.2 Request 154](#_Toc76389459)

[3.2.3 Response Interface 154](#_Toc76389460)

[3.2.4 Sample 155](#_Toc76389461)

[3.2.5 Notice 155](#_Toc76389462)

[3.3 CChex\_TM\_UploadRecord 155](#_Toc76389463)

[3.3.1 Function Description 155](#_Toc76389464)

[3.3.2 Response Interface 156](#_Toc76389465)

[3.3.3 Sample 157](#_Toc76389466)

[3.3.4 Notice 157](#_Toc76389467)

[3.4 CChex\_TM\_DownloadRecordByEmployeeIdAndTime 157](#_Toc76389468)

[3.4.1 Function Description 157](#_Toc76389469)

[3.4.2 Request 157](#_Toc76389470)

[3.4.3 Response Interface 158](#_Toc76389471)

[3.4.4 Sample 159](#_Toc76389472)

[3.4.5 Notice 159](#_Toc76389473)

[3.5 CChex\_GetTRecordNumberByType 159](#_Toc76389474)

[3.5.1 Function Description 159](#_Toc76389475)

[3.5.2 Request 159](#_Toc76389476)

[3.5.3 Response Interface 160](#_Toc76389477)

[3.5.4 Sample 160](#_Toc76389478)

[3.5.5 Notice 160](#_Toc76389479)

[3.6 CChex\_GetTRecordByType 160](#_Toc76389480)

[3.6.1 Function Description 160](#_Toc76389481)

[3.6.2 Request 161](#_Toc76389482)

[3.6.3 Response Interface 161](#_Toc76389483)

[3.6.4 Sample 162](#_Toc76389484)

[3.6.5 Notice 162](#_Toc76389485)

[3.7 CChex\_GetPictureByTRecordIdType 162](#_Toc76389486)

[3.7.1 Function Description 162](#_Toc76389487)

[3.7.2 Response Interface 163](#_Toc76389488)

[3.7.3 Sample 163](#_Toc76389489)

[3.7.4 Notice 164](#_Toc76389490)

[3.8 CChex\_DelPictureByTRecordIdType 164](#_Toc76389491)

[3.8.1 Function Description 164](#_Toc76389492)

[3.8.2 Response Interface 164](#_Toc76389493)

[3.8.3 Sample 165](#_Toc76389494)

[3.8.4 Notice 165](#_Toc76389495)

[3.9 CChex\_DownloadFacePictureModule 165](#_Toc76389496)

[3.9.1 Function Description 165](#_Toc76389497)

[3.9.2 Request 165](#_Toc76389498)

[3.9.3 Response Interface 166](#_Toc76389499)

[3.9.4 Sample 167](#_Toc76389500)

[3.9.5 Notice 167](#_Toc76389501)

[3.10 CChex\_UploadFacePictureModule 167](#_Toc76389502)

[3.10.1 Function Description 167](#_Toc76389503)

[3.10.2 Response Interface 168](#_Toc76389504)

[3.10.3 Sample 169](#_Toc76389505)

[3.10.4 Notice 169](#_Toc76389506)

[3.11 CCHex\_AddFingerprintOnline\_FacePicture 169](#_Toc76389507)

[3.11.1 Function Description 169](#_Toc76389508)

[3.11.2 Request 169](#_Toc76389509)

[3.11.3 Response Interface 170](#_Toc76389510)

[3.11.4 Sample 170](#_Toc76389511)

[3.11.5 Notice 171](#_Toc76389512)

[4 Return value 171](#_Toc76389513)

# Overview

## Introduction

This API document mainly defines the development interface of the Anviz SDK, which provides reference for developers to connect and operate related devices.

This document is mainly related to the software development engineers, software testing engineers and project managers.

## Overview of functions

This API document mainly provides some interfaces of the client/server to the PC end, getting the parameters of the time attendance/access control, and setting the configuration and parameters of the time attendance/access control device. In order to support multiple devices, the interfaces between queries and settings exist asynchronously. SDK is compiled into the form of a dynamic library.

## System logic structure

Device

PC

CChex\_Init

CChex\_Start

CChex\_Stop

Start

CChex\_GetNetConfig

CChex\_SetNetConfig

CChex\_Update

SDK

SetNetwork

GetNetwork

Result

Stop

## Interface functions

1.4.1 **General Interface**

uint CChex\_Version();

void CChex\_Init();

IntPtr CChex\_Start();

void CChex\_Stop(IntPtr CchexHandle);

int CChex\_Update(IntPtr CchexHandle, int[] DevIdx, int[] Type, IntPtr Buff, int Len);

int CChex\_GetNetConfig(IntPtr CchexHandle, int DevIdx);

int CChex\_SetNetConfig(IntPtr CchexHandle, int DevIdx, ref CCHEX\_NETCFG\_INFO\_STRU Config);

int CChex\_MsgGetByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);

int CChex\_MsgDelByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);

int CChex\_MsgAddNew(IntPtr CchexHandle, int DevIdx, byte[] Data, int Len);

int CChex\_MsgGetAllHead(IntPtr CchexHandle, int DevIdx);

int CChex\_RebootDevice(IntPtr CchexHandle, int DevIdx);

int CChex\_SetTime(IntPtr CchexHandle, int DevIdx, int Year, int Month, int Day, int Hour, int Min, int Sec);

int CChex\_GetSNConfig(IntPtr CchexHandle, int DevIdx);

int CChex\_DownloadAllRecords(IntPtr CchexHandle, int DevIdx);

int CChex\_DeleteRecordInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_RECORD\_INFO\_STRU Config);

int CChex\_GetBasicConfigInfo(IntPtr CchexHandle, int DevIdx);

int CChex\_SetBasicConfigInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_BASIC\_CFG\_INFO\_STRU Config);

int CChex\_ListPersonInfo(IntPtr CchexHandle, int DevIdx);

int CChex\_ModifyPersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_RET\_PERSON\_INFO\_STRU personlist, byte person\_num);

int CChex\_DeletePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_PERSON\_INFO\_STRU Config);

int CChex\_DownloadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx);

int CChex\_UploadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx, byte[] FingerData, int DataLen);

int CChex\_GetTime(IntPtr CchexHandle, int DevIdx);//(24)

int CChex\_InitUserArea(IntPtr CchexHandle, int DevIdx);

int CChex\_InitSystem(IntPtr CchexHandle, int DevIdx);

int CChex\_GetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx);

int CChex\_SetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx, ref CCHEX\_GET\_BASIC\_CFG\_INFO2\_STRU\_EXT\_INF config);

int CChex\_GetPeriodTime(IntPtr CchexHandle, int DevIdx, byte SerialNumbe);

int CChex\_SetPeriodTime(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_PERIOD\_TIME\_STRU\_EXT\_INF config);

int CChex\_GetTeamInfo(IntPtr CchexHandle, int DevIdx, byte TeamNumbe);

int CChex\_SetTeamInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_TEAM\_INFO\_STRU\_EXT\_INF config);

int CCHex\_AddFingerprintOnline(IntPtr CchexHandle, int DevIdx, ref CCHEX\_ADD\_FINGERPRINT\_ONLINE\_STRU\_EXT\_INF Param);

int CCHex\_ForcedUnlock(IntPtr CchexHandle, int DevIdx, ref CCHEX\_FORCED\_UNLOCK\_STRU\_EXT\_INF Param);

int CCHex\_Udp\_Search\_Dev(IntPtr CchexHandle);

int CCHex\_Udp\_Set\_Dev\_Config(IntPtr CchexHandle, ref CCHEX\_UDP\_SET\_DEV\_CONFIG\_STRU\_EXT\_INF config);

int CCHex\_ClientConnect(IntPtr CchexHandle, byte[] Ip, int Port);

int CCHex\_ClientDisconnect(IntPtr CchexHandle, int DevIdx);

### 1.4.2 Face Deep Series Temperature detection and template

int CChex\_TM\_DownloadAllRecords(IntPtr CchexHandle, int DevIdx);

int CChex\_TM\_DownloadAllNewRecords(IntPtr CchexHandle, int DevIdx);

int CChex\_TM\_UploadRecord(IntPtr CchexHandle, int DevIdx, ref CCHEX\_TM\_UPLOAD\_RECORD\_INFO\_STRU Record);

int CChex\_TM\_DownloadRecordByEmployeeIdAndTime(IntPtr CchexHandle, int DevIdx, ref CCHEX\_GET\_RECORD\_INFO\_BY\_TIME Param);

int CChex\_GetTRecordNumberByType(IntPtr CchexHandle, int DevIdx, int RecordTpye);

int CChex\_GetTRecordByType(IntPtr CchexHandle, int DevIdx, int RecordTpye);

int CChex\_GetPictureByTRecordIdType(IntPtr CchexHandle, int DevIdx, ref CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU Param);

int CChex\_DelPictureByTRecordIdType(IntPtr CchexHandle, int DevIdx, ref CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU Param);

int CChex\_DownloadFacePictureModule(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_PERSON\_INFO\_STRU Param);

int CChex\_UploadFacePictureModule(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_PERSON\_INFO\_STRU Param, byte[] PictureBuff, int BuffLen);

int CCHex\_AddFingerprintOnline\_FacePicture(IntPtr CchexHandle, int DevIdx, ref CCHEX\_ADD\_FINGERPRINT\_ONLINE\_STRU\_EXT\_INF Param);

## Message Type

CChex\_Update(anviz\_handle, dev\_idx, Type, pBuff, len);

You need to loop the query to see if there is an asynchronous return, and if you have a different Type of data parsing, get the data you want.public enum MsgType:int

{

CCHEX\_RET\_RECORD\_INFO\_TYPE = 1

, CCHEX\_RET\_DEV\_LOGIN\_TYPE

, CCHEX\_RET\_DEV\_LOGOUT\_TYPE

, CCHEX\_RET\_DLFINGERPRT\_TYPE = 4

, CCHEX\_RET\_ULFINGERPRT\_TYPE

, CCHEX\_RET\_MODIFY\_PERSON\_INFO\_TYPE = 8

, CCHEX\_RET\_LIST\_PERSON\_INFO\_TYPE

, CCHEX\_RET\_MSGGETBYIDX\_INFO\_TYPE = 12

, CCHEX\_RET\_MSGGETBYIDX\_UNICODE\_INFO\_TYPE

, CCHEX\_RET\_MSGADDNEW\_INFO\_TYPE

, CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

, CCHEX\_RET\_MSGDELBYIDX\_INFO\_TYPE // 15

, CCHEX\_RET\_MSGGETALLHEAD\_INFO\_TYPE

, CCHEX\_RET\_REBOOT\_TYPE

, CCHEX\_RET\_DEV\_STATUS\_TYPE

, CCHEX\_RET\_MSGGETALLHEADUNICODE\_INFO\_TYPE

, CCHEX\_RET\_SETTIME\_TYPE // 20

, CCHEX\_RET\_UPLOADFILE\_TYPE

, CCHEX\_RET\_GETNETCFG\_TYPE

, CCHEX\_RET\_SETNETCFG\_TYPE

, CCHEX\_RET\_GET\_SN\_TYPE //24

, CCHEX\_RET\_GET\_BASIC\_CFG\_TYPE = 29

, CCHEX\_RET\_SET\_BASIC\_CFG\_TYPE

, CCHEX\_RET\_DEL\_PERSON\_INFO\_TYPE = 31

, CCHEX\_RET\_DEL\_RECORD\_OR\_FLAG\_INFO\_TYPE = 33

,CCHEX\_RET\_GET\_BASIC\_CFG2\_TYPE = 37

,CCHEX\_RET\_SET\_BASIC\_CFG2\_TYPE = 38

,CCHEX\_RET\_GETTIME\_TYPE = 39

,CCHEX\_RET\_INIT\_USER\_AREA\_TYPE = 40

,CCHEX\_RET\_INIT\_SYSTEM\_TYPE = 41

,CCHEX\_RET\_GET\_PERIOD\_TIME\_TYPE = 42

,CCHEX\_RET\_SET\_PERIOD\_TIME\_TYPE = 43

,CCHEX\_RET\_GET\_TEAM\_INFO\_TYPE = 44

,CCHEX\_RET\_SET\_TEAM\_INFO\_TYPE = 45

,CCHEX\_RET\_ADD\_FINGERPRINT\_ONLINE\_TYPE = 46

,CCHEX\_RET\_FORCED\_UNLOCK\_TYPE = 47

,CCHEX\_RET\_UDP\_SEARCH\_DEV\_TYPE = 48

,CCHEX\_RET\_UDP\_SET\_DEV\_CONFIG\_TYPE = 49

,CCHEX\_RET\_GET\_INFOMATION\_CODE\_TYPE = 50

,CCHEX\_RET\_SET\_INFOMATION\_CODE\_TYPE = 51

,CCHEX\_RET\_GET\_BELL\_INFO\_TYPE = 52

,CCHEX\_RET\_SET\_BELL\_INFO\_TYPE = 53

,CCHEX\_RET\_LIVE\_SEND\_ATTENDANCE\_TYPE = 54

,CCHEX\_RET\_GET\_USER\_ATTENDANCE\_STATUS\_TYPE = 55

,CCHEX\_RET\_SET\_USER\_ATTENDANCE\_STATUS\_TYPE = 56

,CCHEX\_RET\_CLEAR\_ADMINISTRAT\_FLAG\_TYPE = 57

,CCHEX\_RET\_GET\_SPECIAL\_STATUS\_TYPE = 58

,CCHEX\_RET\_GET\_ADMIN\_CARD\_PWD\_TYPE = 59

,CCHEX\_RET\_SET\_ADMIN\_CARD\_PWD\_TYPE = 60

,CCHEX\_RET\_GET\_DST\_PARAM\_TYPE = 61

,CCHEX\_RET\_SET\_DST\_PARAM\_TYPE = 62

,CCHEX\_RET\_GET\_DEV\_EXT\_INFO\_TYPE = 63

,CCHEX\_RET\_SET\_DEV\_EXT\_INFO\_TYPE = 64

,CCHEX\_RET\_GET\_BASIC\_CFG3\_TYPE = 65

,CCHEX\_RET\_SET\_BASIC\_CFG3\_TYPE = 66

,CCHEX\_RET\_CONNECTION\_AUTHENTICATION\_TYPE = 67

,CCHEX\_RET\_GET\_RECORD\_NUMBER\_TYPE = 68

,CCHEX\_RET\_GET\_RECORD\_BY\_EMPLOYEE\_TIME\_TYPE = 69

,CCHEX\_RET\_GET\_RECORD\_INFO\_STATUS\_TYPE = 70

,CCHEX\_RET\_GET\_NEW\_RECORD\_INFO\_TYPE = 71

,CCHEX\_RET\_ULEMPLOYEE2W2\_INFO\_TYPE = 72,

CCHEX\_RET\_GET\_BASIC\_CFG5\_TYPE = 73,

CCHEX\_RET\_SET\_BASIC\_CFG5\_TYPE = 74,

CCHEX\_RET\_GET\_CARD\_ID\_TYPE = 75,

CCHEX\_RET\_SET\_DEV\_CURRENT\_STATUS\_TYPE = 76,

CCHEX\_RET\_GET\_URL\_TYPE = 77,

CCHEX\_RET\_SET\_URL\_TYPE = 78,

CCHEX\_RET\_GET\_STATUS\_SWITCH\_TYPE = 79,

CCHEX\_RET\_SET\_STATUS\_SWITCH\_TYPE = 80,

CCHEX\_RET\_GET\_STATUS\_SWITCH\_EXT\_TYPE = 81,

CCHEX\_RET\_SET\_STATUS\_SWITCH\_EXT\_TYPE = 82,

CCHEX\_RET\_UPDATEFILE\_STATUS\_TYPE = 83,

CCHEX\_RET\_GET\_MACHINE\_ID\_TYPE = 84,

CCHEX\_RET\_SET\_MACHINE\_ID\_TYPE = 85,

CCHEX\_RET\_GET\_MACHINE\_TYPE\_TYPE = 86,

CCHEX\_RET\_UPLOAD\_RECORD\_TYPE = 87,

CCHEX\_RET\_GET\_ONE\_EMPLOYEE\_INFO\_TYPE = 88,

CCHEX\_RET\_ULEMPLOYEE\_VER\_4\_NEWID\_TYPE = 89,

CCHEX\_RET\_MANAGE\_LOG\_RECORD\_TYPE = 90,

CCHEX\_RET\_PICTURE\_GET\_TOTAL\_NUMBER\_TYPE = 91,

CCHEX\_RET\_PICTURE\_GET\_ALL\_HEAD\_TYPE = 92,

CCHEX\_RET\_PICTURE\_GET\_DATA\_BY\_EID\_TIME\_TYPE = 93,

CCHEX\_RET\_PICTURE\_DEL\_DATA\_BY\_EID\_TIME\_TYPE = 94,

CCHEX\_RET\_LIVE\_SEND\_SPECIAL\_STATUS\_TYPE = 95,

CCHEX\_RET\_TM\_ALL\_RECORD\_INFO\_TYPE = 150, //CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

CCHEX\_RET\_TM\_NEW\_RECORD\_INFO\_TYPE = 151, //CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

CCHEX\_RET\_TM\_LIVE\_SEND\_RECORD\_INFO\_TYPE = 152, //CCHEX\_RET\_TM\_LIVE\_SEND\_RECORD\_INFO\_STRU

CCHEX\_RET\_TM\_UPLOAD\_RECORD\_INFO\_TYPE = 153, //CCHEX\_RET\_TM\_UPLOAD\_RECORD\_STRU

CCHEX\_RET\_TM\_RECORD\_BY\_EMPLOYEE\_TIME\_TYPE = 154, //CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

CCHEX\_RET\_GET\_T\_RECORD\_NUMBER\_TYPE = 155, //CCHEX\_RET\_GET\_T\_RECORD\_NUMBER\_STRU

CCHEX\_RET\_GET\_T\_RECORD\_TYPE = 156, //CCHEX\_RET\_GET\_T\_RECORD\_STRU

CCHEX\_RET\_GET\_T\_PICTURE\_BY\_RECORD\_ID\_TYPE = 157, //CCHEX\_RET\_GET\_PICTURE\_BY\_RECORD\_ID\_STRU

CCHEX\_RET\_DEL\_T\_PICTURE\_BY\_RECORD\_ID\_TYPE = 158, //CCHEX\_RET\_DEL\_PICTURE\_BY\_RECORD\_ID\_STRU

CCHEX\_RET\_DOWNLOAD\_FACE\_PICTURE\_MODULE\_TYPE = 159, //CCHEX\_DOWNLOAD\_FACE\_PICTURE\_MODULE

CCHEX\_RET\_UPLOAD\_FACE\_PICTURE\_MODULE\_TYPE = 160, //CCHEX\_UPLOAD\_FACE\_PICTURE\_MODULE

CCHEX\_RET\_ADD\_PICTURE\_FINGERPRINT\_ONLINE\_TYPE = 161, //CCHEX\_DOWNLOAD\_FACE\_PICTURE\_MODULE

,CCHEX\_RET\_CLINECT\_CONNECT\_TYPE = 200

,CCHEX\_RET\_CLINECT\_DISCONNECT\_TYPE = 201

CCHEX\_RET\_RECORD\_INFO\_CARD\_BYTE7\_TYPE = 251,

//SAC Door Access Contoller

CCHEX\_SAC\_DOWNLOAD\_COMMON\_TYPE = 401,

CCHEX\_SAC\_UPLOAD\_COMMON\_TYPE = 402,

CCHEX\_SAC\_DELETE\_COMMON\_TYPE = 403,

CCHEX\_SAC\_INIT\_COMMON\_TYPE = 404,

CCHEX\_SAC\_PUSH\_COMMON\_TYPE = 405, };

};

## Setup label and print setup Description

Generic version:

Default Setup: After download new records mark as:0(NO) Print LOG:0(NO)

W2 Version：

Default Setup: After download new records mark as:1(Yes) Print LOG:0(NO)

Customize(S) Version：

Default Setup: After download new records mark as:1(Yes) Print LOG:0(NO)

Customize（F）Version：：

Default Setup: After download new records mark as:0(NO) Print LOG:0(NO)

Change Configuration path:

1：Defining the configuration file：tc-b\_new\_sdk.ini

[LogFile]

LogFile = 1 1：yes 0：no

[SetRecordFlag]

SetRecordFlag = 1 1：yes 0：no

[Debug]

Debug = 0

2:Call the function interface：void CChex\_SetSdkConfig(void \*CchexHandle,int SetRecordflag,int SetLogFile);

Parameter：

int SetRecordflag 1：yes 0：no

int SetLogFile 1：yes 0：no

Priority level:

High： Call the function interface

High： Defining the configuration file

Low： Defining Setup

Notice：When without any “Call the function interface” and “Defining the configuration file as “Defining Setup”

# Description interface

## CChex\_Version

### Description functions

【Function】Query SDK Version.

### Request

【Mode】uint CChex\_Version()

【Parameter】None

### Response

【Return value】Get SDK version, Returns the version number of an integer number.

### Sample

int sdk\_ver = CChex\_Version();

### Notice

## CChex\_Init

### Description functions

【Function】 The SDK function initializes and initializes the socket application interface.

### Request

【Mode】void CChex\_Init()

【Parameter】None

### Response

【Return value】None

### Sample

CChex\_Init();

### Notice

None

## CChex\_Start

### Description functions

【Function】 Start the SDK, allocate space, and establish communication with the Anviz device.

### Request

【Mode】IntPtr CChex\_Start()

【Parameter】None

### Response

【Return value】Return the handle of SDK.

### Sample

IntPtr sdk\_handle = CChex\_Start();

if (sdk\_handle != null)

{

MessageBox.Show("Startup OK");

｝ else

{

MessageBox.Show("Startup errors,Please restart the program.");

}

### Notice

1. Make sure that CChex\_Init is initialized before run.

## CChex\_Stop

### Description functions

【Function】Stop the SDK, release the space, close the socket chain and so on.

### Request

【Mode】void CChex\_Stop(IntPtr CchexHandle)

【Parameter】CchexHandle,CChex\_Start successfully create the handle

### Response

【Return value】None

### Sample

CChex\_stop();

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_Update

### Description functions

【Function】The Return value get or set asynchronously.。

### Request

【Mode】int CChex\_Update(IntPtr CchexHandle, int[] DevIdx, int[] Type, IntPtr Buff, int Len);

【Parameter】CchexHandle,CChex\_Start successfully create the handle，Input[Parameter]；

DevIdx,Device index returned asynchronously,Output[Parameter]；

Buff,Returned data section,Output [Parameter]；

Len,Returns the length of the data section,Input[Parameter]；

### Response

【Return value】> 0：Successful asynchronous ; Return value == 0：Invalid Return value；< 0：buffer space is not enough，based on Return value, then re-apply the space.

### Sample

int ret = CChex\_Update(anviz\_handle, dev\_idx, Type, pBuff, len);

if (ret > 0)

{

switch(Type)

{

case CCHEX\_RET\_DEV\_LOGIN\_TYPE: // Connect the Anviz device successfully and return the Device ID, Device IP address, Software version, Device type and software version, etc.

break;

case CCHEX\_RET\_GET\_BASIC\_CFG\_TYPE: // Some basic information about the Anviz device, such as administrator password, firmware version, volume, etc.

break;

case CCHEX\_RET\_DEV\_STATUS\_TYPE: // Total number of employees, total number of fingerprints, total number of passwords, total number of cards, total attendance records, total new attendance record.

default:

break;

}

}else if (ret == 0)

{

// invalid data

}else

{

// Buff is not enough,

}

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_GetNetConfig

### Description functions

【Function】Query Anviz device’s network Configuration

### Request

【Mode】int CChex\_GetNetConfig(IntPtr CchexHandle, int DevIdx);

【Parameter】CChex\_Start successfully create the handle，Input[Parameter]；

DevIdx,search the device Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GETNETCFG\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte [4]IpAddr;// IP address

byte [4]IpMask;// Mask

byte [6]MacAddr;// MAC

byte [4]GwAddr;// Gateway

byte [4]ServAddr;// Sever IP address

byte RemoteEnable;//Standby

byte [2]Port;// Port

byte Mode;// Mode：0: Server，1:Client

byte DhcpEnable;// DHCP，0:Disable，1:Enable

### Sample

int ret = CChex\_GetNetConfig();

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_SetNetConfig

### Description functions

【Function】Set the network configuration of the Anviz device.

### Request

【Mode】int CChex\_SetNetConfig(IntPtr CchexHandle, int DevIdx, ref CCHEX\_NETCFG\_INFO\_STRU Config);

【Parameter】CchexHandle：,CChex\_Start successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config:

byte [4]IpAddr;// IP Address

byte [4]IpMask;// Mask

byte [6]MacAddr;// MAC address

byte [4]GwAddr;// Gateway

byte [4]ServAddr;// Server IP

byte RemoteEnable;//Standby

byte [2]Port;// Port

byte Mode;// Mode：0:Server，1:Client

byte DhcpEnable;// DHCP，0:Disable，1:Enable

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update,Type return CCHEX\_RET\_SETNETCFG\_TYPE

Data part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = (anviz\_handle, dev\_idx, ref dev\_info);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_MsgGetByIdx

### Description functions

【Function】Reads the designated short message’s start date, end date, and message content .

### Request

【Mode】int CChex\_MsgGetByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);

【Parameter】CchexHandle：,CChex\_Start successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Idx,Read the short message, Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGGETBYIDX\_UNICODE\_INFO\_TYPE

Data part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

int Len; // The length of the following data

byte[5] EmployeeId; // User ID，0 means public short message

byte StartYear; // Start time Year

byte StartMonth; // Start time Month

byte StartDay; // Start time Day

byte EndYear; // End time Year

byte EndMonth; // End time Month

byte EndDay; // End time Day

byte [48\*2]content // Content of short message,UNICODE

### Sample

int ret = CChex\_MsgGetByIdx(anviz\_handle, dev\_idx, msg\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_MsgDelByIdx

### Description functions

【Function】Delete the designated short message

### Request

【Mode】int CChex\_MsgDelByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);

【Parameter】CChex\_Start， successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Idx,Delete the designed short message,Input[Parameter]

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGDELBYIDX\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret =CChex\_MsgDelByIdx(anviz\_handle, dev\_idx, msg\_idx);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_MsgAddNew

### Description functions

【Function】Add short message.

### Request

【Mode】int CChex\_MsgAddNew(IntPtr CchexHandle, int DevIdx, byte[] Data, int Len);

【Parameter】CChex\_Start，successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Data,Short message, Input[Parameter]；

byte[5] EmployeeId; // User ID，0 means public short message

byte StartYear; // Start time Year

byte StartMonth; // Start time Month

byte StartDay; // Start time Day

byte EndYear; // End time Year

byte EndMonth; // End time Month

byte EndDay; // End time Day

byte [48\*2]content // Content of short message,UNICODE

Len,the length of short message，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_MsgAddNew(anviz\_handle, dev\_idx, send\_buff, send\_len);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_MsgGetAllHead

### Description functions

【Function】Read all short message headers.

### Request

【Mode】int CChex\_MsgGetAllHead(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

int Len; // All 50 SMS users, start and end time.

byte[5] EmployeeId; // User ID，0 means public SMS

byte StartYear; // Start time Year

byte StartMonth; // Start time Month

byte StartDay; // Start time Day

byte EndYear; // End time Year

byte EndMonth; // End time Month

byte EndDay; // End time Day

...Total 50 records

### Sample

int ret =CChex\_MsgGetAllHead(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.已经成功启动；

## CChex\_RebootDevice

### Description functions

【Function】Restart the Anviz device

### Request

【Mode】int CChex\_RebootDevice(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_RebootDevice(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.已经成功启动；

## CChex\_SetTime

### Description functions

【Function】Set date/time of the Anviz device

### Request

【Mode】int CChex\_SetTime(IntPtr CchexHandle, int DevIdx, int Year, int Month, int Day, int Hour, int Min, int Sec);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Year、Month、Day、Hour、 Min、 Sec，Input[Parameter]

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_SetTime(anviz\_handle, dev\_idx, 2018, 1, 25, 10, 10, 10);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.已经成功启动；

## CChex\_GetSNConfig

### Description functions

【Function】Get the SN of the Anviz device

### Request

【Mode】int CChex\_GetSNConfig(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]句柄，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte[16] sn; // SN，Each number, and it needs to convert into characters

### Sample

int ret = CChex\_GetSNConfig(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_DownloadAllRecords

### Description functions

【Function】Download all attendance records.

### Request

【Mode】int CChex\_DownloadAllRecords(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE Data Part：

uint MachineId; // Device ID

byte NewRecordFlag; // New or not record?

byte[5] EmployeeId; // User ID，

byte[4] Date; // date/time，The number of seconds apart from 2000.

byte BackId; // Backup ID，the 3 digits; Card，the 2 digits; Password，the 1~0; 11，7~4 digits; 10 Fingerprints（1~10）digits

byte RecordType // Record type，7 digits; 1：open the door ，0：can’t open the door ；3~0 digits attendance status

byte[3] WorkType; // Workcode

byte Rsv // Reserved

### Sample

int ret = CChex\_DownloadAllRecords(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_DeleteRecordInfo

### Description functions

【Function】Delete all records, or clear all/ some new records flag.

### Request

【Mode】int CChex\_DeleteRecordInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_RECORD\_INFO\_STRU Config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config,Input[Parameter]；

byte del\_type; // 0:Delete all records；1：Clear all new record flag；2：Clear some records flag，and the deleted number based on del\_count

uint del\_count; // del\_type =2时，assigned clear new records number

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

uint deleted\_count; // How many records or new records flag be deleted?

### Sample

int ret = CChex\_DeleteRecordInfo(anviz\_handle, dev\_idx, ref delete\_record);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_GetBasicConfigInfo

### Description functions

【Function】Get the basic configuration of the Anviz device

### Request

【Mode】int CChex\_GetBasicConfigInfo(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_GET\_BASIC\_CFG\_INFO\_STRU Cfg;

byte[8] software\_version; // firmware version，

uint password; // Password，longest support 6 digits

byte delay\_for\_sleep; // Delay sleep time，0~250 Mins，0 don’t sleep

byte volume; // Volume，0~5，0：mute，5：Max byte language; //Language，0：Simplified Chinese，1：Traditional Chinese，2：English，3.French，4：Spanish，5：Portuguese

byte date\_format; // Date format，0:Chinese，1：UK，2：USA byte time\_format; // Time format，0：24Houes，1：12Hours

byte machine\_status; // Attendance Status，0~15

byte modify\_language; // Modify Language，0x10 You can modify the language of the device.

byte cmd\_version; // Instruction version

### Sample

int ret = CChex\_GetBasicConfigInfo(anviz\_handle, dev\_idx);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.；

## CChex\_SetBasicConfigInfo

### Description functions

【Function】Set the basic configuration of the Anviz device

### Request

【Mode】int CChex\_SetBasicConfigInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_BASIC\_CFG\_INFO\_STRU Config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config,Input[Parameter]；Unmodified fields，0xFF

uint password; // Password，String convert into number

byte pwd\_len; // Password length

byte delay\_for\_sleep; // Delay sleep time，0~250 Mins，0 don’t sleep

byte volume; // Volume，0~5，0：mute，5：Max

byte language; // Language，0：Simplified Chinese，1：Traditional Chinese，2：English，3.French，4：Spanish，5：Portuguese

byte date\_format; // Date format，0:Chinese，1：UK，2：USA

byte time\_format; // Time format，0：24Houes，1：12Hours

byte machine\_status; // Attendance Status，0~15

byte modify\_language; // Modify Language，0x10 You can modify the language of the device.

byte rsv; // Reserved

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_SetBasicConfigInfo(anviz\_handle, dev\_idx, ref set\_basic\_cfg);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_ListPersonInfo

### Description functions

【Function】Get all user information.

### Request

【Mode】int CChex\_ListPersonInfo(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int CurIdx; // Current index number

int TotalCnt; // The total number of users

byte [5]EmployeeId; // User ID，5 bytes,Longest 12digits

byte password\_len; // Password length

byte max\_password; // Maximum length of password,= 6，unchangeable

int password; // Password

byte max\_card\_id; // The maximum length of the card number，= 6（3bytes）or 10 （4bytes）, unchangeable

uint card\_id; // Card number

byte max\_EmployeeName; // The maximum length of a user's name，= 10、20 、64 and160,unchangeable；=160 ,The user name is stored in EmployeeName2.

byte[64] EmployeeName; // User name

byte DepartmentId; // Department ID

byte GroupId; // Group ID

byte Mode; // Attendance mode

uint Fp\_Status; // Fingerprint register status，0~9:fp; 10:face; 11:iris1; 12:iris2

byte Rserved1; // for 22

byte Rserved2; // for 72 and 22

byte Special; // Special information

// DR info

byte[160] EmployeeName2; // Employee Name2

byte[13] RFC; // RFC Information

byte[18] CURP; // CURP Information

### Sample

int ret = CChex\_ListPersonInfo(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_ModifyPersonInfo

### Description functions

【Function】Modify the relevant user’s information.

### Request

【Mode】int CChex\_ModifyPersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_RET\_PERSON\_INFO\_STRU personlist, byte person\_num);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

person\_num，How many users were changed，Input[Parameter]；

Personlist，user details information，Input[Parameter]；

uint MachineId; // Device ID

int CurIdx; // Current index number

int TotalCnt; // The total number of users

byte [5]EmployeeId; // User ID，5 bytes,Longest 12digits

byte password\_len; // Password length

byte max\_password; // Maximum length of password,= 6，unchangeable

int password; // Password

byte max\_card\_id; // The maximum length of the card number，= 6（3bytes）or 10 （4bytes）, unchangeable

uint card\_id; // Card number

byte max\_EmployeeName; // The maximum length of a user's name，= 10、20 、64 and160,unchangeable；=160 ,The user name is stored in EmployeeName2.

byte[64] EmployeeName; // User name

byte DepartmentId; // Department ID

byte GroupId; // Group

byte Mode; // Attendance mode

uint Fp\_Status; // Fingerprint register status，0~9:fp; 10:face; 11:iris1; 12:iris2

byte Rserved1; // for 22

byte Rserved2; // for 72 and 22

byte Special; // Special information

// DR info

byte[160] EmployeeName2; // User name 2

byte[13] RFC; // RFC Information

byte[18] CURP; // CURP Information

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_ModifyPersonInfo(anviz\_handle, dev\_idx, ref item, 1);

### Notice

1.Make sure that CChex\_Start has been started successfully before run；

## CChex\_DeletePersonInfo

### Description functions

【Function】Deletes the assigned person’s information.

### Request

【Mode】int CChex\_DeletePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_PERSON\_INFO\_STRU Config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config,Input[Parameter]；

byte []EmployeeId; // User ID

byte operation; // Which information be deleted ? The 3 digits: Card; The 2 digits: Password; The 1~0 digits: 1~0 digits ：11; 7~4 digits：10 Fingerprints（1~10）；0xFF: Delete all user’s information .

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_DeletePersonInfo(anviz\_handle, dev\_idx, ref delete\_item);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_DeletePersonInfo\_VER\_4\_NEWID

### Description functions

【Function】Delete the user’s information

(Version:User ID for Chat DevTypeFlag & 0XFF ==DEV\_TYPE\_VER\_4\_NEWID)

### Request

【Mode】int CChex\_DeletePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_EMPLOYEE\_INFO\_STRU\_EXT\_INF\_ID\_VER\_4\_NEWID Config);

【Parameter】CchexHandle：successfully create by CChex\_Start the handle，Input[Parameter]；；

DevIdx: Index of device，Input[Parameter]；

Config: Input[Parameter]；

byte [28]EmployeeId; // User ID for the String

byte operation; // Delete user Information the 3 bit is card, The 2 bit is Password ,The 0~1 byte is 11 and the 7~14 bit is number of the fingerprints(1~10)； 0xFF，delete all user information

### Response

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_DEL\_EMPLOYEE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte [28] EmployeeId; // User ID for the String

### Sample

int ret = CChex\_DeletePersonInfo(anviz\_handle, dev\_idx, ref delete\_item);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_DownloadFingerPrint

### Description functions

【Function】Download fingerprint template from Anviz device

### Request

【Mode】int CChex\_DownloadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx,Search the device，Input[Parameter]；

EmployeeId，User ID, Input[Parameter]；

FingerIdx，Fingerprint Indexing，1~10：means fingerprint number 1~10，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte []EmployeeId; // User ID

byte FpIdx; // Fingerprint indexing，1~10：means fingerprint number 1~10

uint fp\_data\_len; // The length of the fingerprint data.

byte []Data; // fingerprint data

### Sample

int ret = CChex\_DownloadFingerPrint(anviz\_handle, dev\_idx, EmployeeID, FpIdx);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_DownloadFingerPrint\_VER\_4\_NEWID

### Function

【Function】Download the fingerprint information from the device

**(Version:device version DevTypeFlag & 0XFF ==DEV\_TYPE\_VER\_4\_NEWID)**

### Request

【Mode】int CChex\_DownloadFingerPrint\_VER\_4\_NEWID(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

EmployeeId: byte[28]User ID，Input[Parameter]; User ID for the String FingerIdx: Fingerprint indexing，1~10：means fingerprint number 1~10，Input[Parameter]；

### Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_DLFINGERPRT\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte [28]EmployeeId; // User ID //User ID for the String ID byte[28]

byte FpIdx; // Fingerprint indexing，1~10：means fingerprint number 1~10

uint fp\_data\_len; // The length of the fingerprint data.

byte []Data; // fingerprint data

### Sample

int ret = CChex\_DownloadFingerPrint\_VER\_4\_NEWID(anviz\_handle, dev\_idx, EmployeeID, FpIdx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_UploadFingerPrint

### Description functions

【Function】Upload fingerprint template to Anviz device

### Request

【Mode】int CChex\_UploadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx, byte[] FingerData, int DataLen);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

EmployeeId，User ID，Input[Parameter]；

FingerId; Fingerprint indexing，1~10：means fingerprint number 1~10，Input[Parameter]；

FingerData，fingerprint data，Input[Parameter]；

DataLen，The length of the fingerprint data，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_DownloadFingerPrint(anviz\_handle, dev\_idx, EmployeeID, FpIdx);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_UploadFingerPrint\_VER\_4\_NEWID

### Function

【Function】Upload the fingerprint template to device (Device Type:"FACEPASS7" For :FaceTemplate download )

**(Version:Device Version Model DevTypeFlag & 0XFF ==DEV\_TYPE\_VER\_4\_NEWID)**

### Request

【Mode】int CChex\_UploadFingerPrint\_VER\_4\_NEWID(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx, byte[] FingerData, int DataLen);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；；

DevIdx: Index of device，Input[Parameter]；

EmployeeId: User ID，Input[Parameter]; //User ID for the String ID byte[28]

FingerIdx: Fingerprint indexing，1~10：means fingerprint number 1~10，Input[Parameter]；

FingerData，fingerprint data，Input[Parameter]；

DataLen，The length of the fingerprint data，Input[Parameter]；

### Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type return CCHEX\_RET\_ULFINGERPRT\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte []EmployeeId; // /User ID for the String ID byte[28]

byte FpIdx; // Fingerprint indexing，1~10：means fingerprint number 1~10，Input[Parameter]；

uint fp\_data\_len; // The length of the fingerprint data

byte []Data; // Fingerprint Data

### Sample

int ret = CChex\_DownloadFingerPrint\_VER\_4\_NEWID(anviz\_handle, dev\_idx, EmployeeID, FpIdx);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CChex\_GetTime

### Description functions

【Function】Get the date/time of the Anviz device.

### Request

【Mode】int CChex\_GetTime(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_MSG\_GETTIME\_STRU Cfg;

uint Year; //Year

uint Month; //Month

uint Day; //Day

uint Hour; //Hour

uint Min; //Minute

uint Sec; //Second

### Sample

int ret = CChex\_GetTime(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_InitUserArea

### Description functions

【Function】Initial user area

Initialize all user data areas，Clear all user data, fingerprint data, password/card data.

### Request

【Mode】int CChex\_InitUserArea(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_InitUserArea(anviz\_handle, dev\_idx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_InitSystem

### Description functions

【Function】initial device

initialize the Anviz device to restore factory Default.

### Request

【Mode】int CChex\_InitSystem(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_InitSystem(anviz\_handle, dev\_idx);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_GetBasicConfigInfo2

### Description functions

【Function】Get the Anviz device’s configuration2

Get device’s identification precision, fixed wiegand head, wiegand mode, workcode, real-time attendance records , schedule bell , lock control delay, record overflow warning, repeated attendance record time , door sensor delay, schedule bell delay.

### Request

【Mode】int CChex\_GetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_GET\_BASIC\_CFG\_INFO2\_STRU\_EXT\_INF Param;

byte compare\_level; //identification precision

byte wiegand\_range; //fixed wiegand head

byte wiegand\_type; //wiegand mode

byte work\_code; //Workcode function, enable or disable?

byte real\_time\_send; //Realtime function ,enable or disable?

byte auto\_update; //Auto-update function, enable or disable?

byte bell\_lock; //Schedule bell function, enable or disable?

byte lock\_delay; //Lock time delay

uint record\_over\_alarm; //Record overflow alarm

byte re\_attendance\_delay; //Repeat attendance time delay

byte door\_sensor\_alarm; //Door sensor time delay

byte bell\_delay; //Schedule bell time delay

byte correct\_time; //Time calibration

### Sample

int ret = CChex\_GetBasicConfigInfo2(anviz\_handle, dev\_idx);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_SetBasicConfigInfo2

### Description functions

【Function】Set up the Anviz device’s configuration information2

### Request

【Mode】int CChex\_SetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx, ref CCHEX\_GET\_BASIC\_CFG\_INFO2\_STRU\_EXT\_INF config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

config: Input[Parameter]；

byte compare\_level; //identification precision

byte wiegand\_range; //fixed wiegand head

byte wiegand\_type; //wiegand mode

byte work\_code; //Workcode function, enable or disable?

byte real\_time\_send; //Realtime function ,enable or disable?

byte auto\_update; //Auto-update function, enable or disable?

byte bell\_lock; //Schedule bell function, enable or disable?

byte lock\_delay; //Lock time delay

uint record\_over\_alarm; //Record overflow alarm

byte re\_attendance\_delay; //Repeat attendance time delay

byte door\_sensor\_alarm; //Door sensor time delay

byte bell\_delay; //Schedule bell time delay

byte correct\_time; //Time calibration

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_SetBasicConfigInfo2(anviz\_handle, dev\_idx,ref config);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_GetPeriodTime

### Description functions

【Function】Get time zone from device

Read the timezone information，Total 32 timezones。

### Request

【Mode】int CChex\_GetPeriodTime(IntPtr CchexHandle, int DevIdx, byte SerialNumbe);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

SerialNumbe,timezone number，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_GET\_PERIOD\_TIME\_ONE\_STRU\_EXT\_INF day[7]; //One week 7 days,28Byte

byte StartHour; //Start hour

byte StartMin; //Start mins

byte EndHour; //End hour

byte EndMin; //End mins

### Sample

int ret = CChex\_GetPeriodTime(anviz\_handle, dev\_idx,SerialNumbe);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_SetPeriodTime

### Description functions

【Function】Set timezone

Total 32 timezones。

### Request

【Mode】int CChex\_SetPeriodTime(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_PERIOD\_TIME\_STRU\_EXT\_INF config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config: Input[Parameter]；

byte SerialNumbe;

CCHEX\_GET\_PERIOD\_TIME\_ONE\_STRU\_EXT\_INF day[7]; //one week with 7 days,so should be have 7 groups,28Byte

byte StartHour; //Start Hour

byte StartMin; //Start Mins

byte EndHour; //End Hours

byte EndMin; //End Mins

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_SetPeriodTime(anviz\_handle, dev\_idx,ref config);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run；

## CChex\_GetTeamInfo

### Description functions

【Function】Get Group information

Read one group information, group number 0-16, group0 = normal close; group1=normal open. Only read group2 - group16.

### Request

【Mode】int CChex\_GetTeamInfo(IntPtr CchexHandle, int DevIdx, byte TeamNumbe);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

TeamNumbe:Group number,Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte[4] PeriodTimeNumber;//4Group timezone number(refer to 2.29)

### Sample

int ret = CChex\_GetTeamInfo(anviz\_handle, dev\_idx,TeamNumbe);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CChex\_SetTeamInfo

### Description functions

【Function】Setup group

Setup one group, group number 0- 16, group 0= normal close; group1= normal open, only setup group2 -group16.

### Request

【Mode】int CChex\_SetTeamInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_TEAM\_INFO\_STRU\_EXT\_INF config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config: Input[Parameter]；

byte TeamNumbe; //Group Number

byte[4] PeriodTimeNumber; //4Group ,timezone

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CChex\_SetTeamInfo(anviz\_handle, dev\_idx,ref config);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CCHex\_AddFingerprintOnline

### Description functions

【Function】Register fingerprint online

### Request

【Mode】int CCHex\_AddFingerprintOnline(IntPtr CchexHandle, int DevIdx, ref CCHEX\_ADD\_FINGERPRINT\_ONLINE\_STRU\_EXT\_INF Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param: Input[Parameter]；

byte[5] EmployeeId; //User ID

byte BackupNum; //Backup number

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CCHex\_AddFingerprintOnline(anviz\_handle, dev\_idx,ref Param);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.
2. After the recall, the device will register the fingerprint twice, timeout is 10 seconds, the registration error (timeout, operation failure, the user already exists, the fingerprint exists) returned -1, and successfully returns 0.

## CCHex\_ForcedUnlock

### Description functions

【Function】Open the door by software (Force open the lock)

### Request

【Mode】int CCHex\_ForcedUnlock(IntPtr CchexHandle, int DevIdx, ref CCHEX\_FORCED\_UNLOCK\_STRU\_EXT\_INF Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param: Input[Parameter]；( Param =IntPtr.Zerohe ; default null pointer , Malaysia panasonic project customization needs to enter Param);

byte LockCmd; //Lock command

byte[5] EmployeeId; //User ID

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CCHex\_ForcedUnlock(anviz\_handle, DevIdx,IntPtr.Zero);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.；
2. If Not Malaysia Panasonic project customization, Param none pointer..

## CCHex\_Udp\_Search\_Dev

### Description functions

【Function】UDP Research the Anviz device

### Request

【Mode】int CCHex\_Udp\_Search\_Dev(IntPtr CchexHandle);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：(Data length =sizeof(int)+DevNum\*sizeof(CCHEX\_UDP\_SEARCH\_STRU\_EXT\_INF))

int DevNum; // quantity of device, ( =0, No device,data length =4)

CCHEX\_UDP\_SEARCH\_STRU\_EXT\_INF[] dev\_net\_info ; //Device Data(Single 180Byte)

uint MachineId; //Device ID

int Result; //0: ok -1: fail

int DevHardwareType; //device type(0:Normal device;1:support DNS;2:support wifi)

byte[167] Data; //Data(Convert based on device type)

Byte Padding; //Data alignment, padding

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

byte[167] Data: Data type definition

Device type = 0 :Normal device

CCHEX\_UDP\_SEARCH\_STRU BasicSearchInfo; //normal device 63Byte

byte[10] DevType; //device type

byte[16] DevSerialNum; //device serial numebr

byte[4] IpAddr; //IP Address

byte[4] IpMask; //MASK

byte[4] GwAddr; //Gateway

byte[6] MacAddr; //MAC

byte[4] ServAddr; //Server IP

byte[2] Port; //Port

byte NetMode; //network mode

byte[8] Version; //Firmware version

byte[4] Reserved; //Reserved

Device type = 1 :Support DNS firmware

CCHEX\_UDP\_SEARCH\_WITH\_DNS\_STRU DnsSearchInfo; //Support DNS firmware 167Byte

CCHEX\_UDP\_SEARCH\_STRU BasicSearchInfo; //device type 0:normal device 63Byte structure

byte[4] Dns; //DNS

byte[100] Url; //URL

Device type = 2 : Support wifi

CCHEX\_UDP\_SEARCH\_TWO\_CARD\_STRU TwocardSearchInfo; \\ support wifi 102Byte

byte[10] DevType; //Device type

byte[16] DevSerialNum; //Device serial number

byte[4] ServAddr; //Server IP

byte[2] Port; //Port

byte NetMode; //Network mode

byte[8] Version; //Firmware version

byte[4] Reserved; //Reserved

byte CardNumber; //Network card number

CCHEX\_UDP\_SEARCH\_CARD\_STRU CardInfo[2]; //2group network card information: single network card information = 28Byte

byte[10] CardName; //Network card name

byte[4] IpAddr; //IP Address

byte[4] IpMask; //MASK

byte[4] GwAddr; //Gateway

byte[6] MacAddr; //MAC Address

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

### Sample

int ret = CCHex\_Udp\_Search\_Dev(anviz\_handle);

### Notice

1. Make sure that CChex\_Start has been started successfully before run.

## CCHex\_Udp\_Set\_Dev\_Config

### Description functions

【Function】UDP setting device configuration

### Request

【Mode】int CCHex\_Udp\_Set\_Dev\_Config(IntPtr CchexHandle, ref CCHEX\_UDP\_SET\_DEV\_CONFIG\_STRU\_EXT\_INF config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

Config: Input[Parameter]；

byte[4] IpAddr; //IP Address

byte[4] IpMask; //MASK

byte[4] GwAddr; //Gateway

byte[6] MacAddr; //MAC address

byte[4] ServAddr; //Server IP

byte[2] Port; //Port

byte NetMode; //Network mode

byte[3] Padding; //structure alignment,padding 3byte

uint NewMachineId; //New Device ID

byte[4] Reserved; //Reserved

byte[12] DevUserName; //User name

byte[12] DevPassWord; //Password

int DevHardwareType; //Device type 0:Normal device ;1:support DNS device

byte[4] Dns; //DNS device type = 0 Dns , unchangeable

byte[100] Url; //URL device type = 0 Url,unchangeable

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### Sample

int ret = CCHex\_Udp\_Set\_Dev\_Config(anviz\_handle,ref config);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CCHex\_ClientConnect

### Description functions

【Function】The client actively connects to the server.

### Request

【Mode】int CCHex\_ClientConnect(IntPtr CchexHandle, byte[] Ip, int Port);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

Ip: Server IP address，Input[Parameter]；

Port: Server Port, Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】Failure:CChex\_Update,Type return CCHEX\_RET\_CLINECT\_CONNECT\_TYPE;

Data Part：

int Result; //-1 err

byte[24] Addr; //IP:Port 如(192.168.100.100:5010)

Successful:CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

int DevIdx; //

uint MachineId; //Device ID

byte[24] Addr; //IP:Port For example(192.168.100.100:5010)

byte[8] Version; //Version Number

byte[8] DevType; //Device type

int DevTypeFlag; //Type flag

### Sample

int ret = CCHex\_ClientConnect(anviz\_handle,Ip,Port);

### Notice

1. 1.Make sure that CChex\_Start has been started successfully before run.

## CCHex\_ClientDisconnect

### Description functions

【Function】The client actively disconnects the server.

### Request

【Mode】int CCHex\_ClientDisconnect(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】Failure:CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

int Result; // -1:error

int DevIdx; //

Successful:CChex\_Update, Type return CCHEX\_RET\_MSGADDNEW\_UNICODE\_INFO\_TYPE

Data Part：

int DevIdx; //Index of device，Input[Parameter]；

uint MachineId; //Device ID

uint Live; //

byte[24] Addr; //ip:Port 如(192.168.100.100:5010)

byte[8] Version; //Version number

byte[8] DevType; //Device type

### Sample

int ret = CCHex\_ClientDisconnect(anviz\_handle,DevIdx);

### Notice

1.Make sure that CChex\_Start has been started successfully before run.

## 2.42 CChex\_GetInfomationCode

### 2.42.1 Description function

【Function】Gets the factory information

Read the factory information ANSI version data:10Byte,UNICODE version data:20Byte

### 2.42.2 Request

【Mode】int CChex\_GetInfomationCode(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.40.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_INFOMATION\_CODE\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

int fp\_len; //ANSI VERSION:fp\_len == 10 UNICODE VERSION:fp\_len == 20

byte[20] data; //

### 2.42.4 Sample

int ret = CChex\_GetInfomationCode(anviz\_handle, dev\_idx);

### 2.42.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.43 CChex\_SetInfomationCode

### 2.43.1 Description function

【Function】Modify the factory information

Modify the factory information ANSI version data:10Byte ,UNICODE version data:20Byte

### 2.43.2 Request

【Mode】int CChex\_SetInfomationCode(IntPtr CchexHandle, int DevIdx ,byte[] Data, int DataLen);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Data: Byte[10] or Byte[20] parameter，Input[Parameter]；

DataLen: Write the parameters based on the version, , ANSI version = 10,UNICODE version = 20, Input[Parameter]；

### 2.43.3Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_INFOMATION\_CODE\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.43.4 Sample

int ret = CChex\_SetInfomationCode(anviz\_handle, dev\_idx,Data,DataLen);

### 2.43.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.44 CChex\_GetBellInfo

### 2.44.1 Description function

【Function】Get the Schedule bell information

Read all time of schedule bell，totally 30.

### 2.44.2 Request

【Mode】int CChex\_GetBellInfo(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.44.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_BELL\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_RET\_GET\_BELL\_TIME\_POINT time\_point[30];

byte hour;

byte minute;

byte flag\_week; //Week mark/flag(Binary 0000000 means：Sat Fri Thu Wed Tue Mon Sun)

Byte[2] padding; //data structure alignment , invalid data

### 2.44.4 Sample

int ret = CChex\_GetBellInfo(anviz\_handle, dev\_idx);

### 2.44.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.45 CChex\_SetBellInfo

### 2.45.1 Description function

【Function】Setup the schedule bell information

Setup the time of schedule bell

### 2.45.2 Request

【Mode】int CChex\_SetBellInfo(IntPtr CchexHandle, int DevIdx ,byte BellTimeNum,byte Hour,byte Min,byte FlagWeek);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

BellTimeNum：The time to ring the bell，Input[Parameter]；

Hour：Hour，Input[Parameter]；

Min：Minute，Input[Parameter]；

FlagWeek：Week mark/flag(Binary 0000000means：Sat Fri Thu Wed Tue Mon Sun)，Input[Parameter]；

### 2.45.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_BELL\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.45.4 Sample

int ret = CChex\_SetBellInfo(CchexHandle,DevIdx ,BellTimeNum,Hour,Min,FlagWeek);

### 2.45.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.46 CChex\_GetUserAttendanceStatusInfo

### 2.46.1 Description function

【Function】Gets the self-defining attendance status table

Read the information of self-defining attendance status.

### 2.46.2 Request

【Mode】int CChex\_GetUserAttendanceStatusInfo(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.46.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_USER\_ATTENDANCE\_STATUS\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

int fp\_len; //ANSI VERSION：fp\_len == 80 UNICODE VERSION fp\_len == 160

Byte atten\_status\_number; //Number of attendance status == 8 default

Byte[160] data\_info; //data format: ANSI VERSION: unsigned char [8][10] UNICODE VERSION: unsigned char[8][20] 。

Byte[3] padding; //data structure alignment, invalid data

### 2.46.4 Sample

int ret = CChex\_GetUserAttendanceStatusInfo(anviz\_handle, dev\_idx);

### 2.46.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.47 CChex\_SetUserAttendanceStatusInfo

### 2.47.1 Description function

【Function】Setup the self-defining attendance status

Setup the self-defining attendance status.

### 2.47.2 Request

【Mode】int CChex\_SetUserAttendanceStatusInfo(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_USER\_ATTENDANCE\_STATUS\_STRU Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：8 groups self-defining status,Input[Parameter]；

uint fp\_len;//ANSI VERSION fp\_len = 80 UNICODE VERSION fp\_len = 160

Byte atten\_status\_number; //Number of attendance status 8 default

Byte[160] data\_info;//ANSI VERSION:8][10]，UNICODE VERSION:[8][20]

Byte[3] padding; //alignment, invalid data

### 2.47.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_USER\_ATTENDANCE\_STATUS\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.47.4 Sample

int ret = CChex\_SetUserAttendanceStatusInfo(CchexHandle,DevIdx ,ref Param);

### 2.47.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.48 CChex\_ClearAdministratFlag

### 2.48.1 Description function

【Function】Clear the administrator flag

Clear all administrator flag.

### 2.48.2 Request

【Mode】int CChex\_ClearAdministratFlag(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.48.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_CLEAR\_ADMINISTRAT\_FLAG\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.48.4 Sample

int ret = CChex\_ClearAdministratFlag(anviz\_handle, dev\_idx);

### 2.48.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.49 CChex\_GetSpecialStatus

### 2.49.1 Description function

【Function】Get the special status

Get the current special status. Only for VF30/VP30/T60+

### 2.49.2 Request

【Mode】int CChex\_GetSpecialStatus(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.49.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_SPECIAL\_STATUS\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

Byte status; //Byte1：Door alarm status 0-normal ,status 1-alarm status; Byte5：Door status 0-close , 1-open; Byte6：Door sensor status 0-close, 1-open; Byte7：Lock status 0-close, 1-open

Byte[7] reserved; //reserved useless in the temporary

### 2.49.4 Sample

int ret = CChex\_GetSpecialStatus(anviz\_handle, dev\_idx);

### 2.49.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.50 CChex\_GetAdminCardnumberPassword

### 2.50.1 Description function

【Function】Read administrator card number or password. Only for T5/M5/T50m

Get administrator card number or password.

### 2.50.2 Request

【Mode】int CChex\_GetAdminCardnumberPassword(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.50.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_ADMIN\_CARD\_PWD\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

Byte[13] data; //device type : T5A,T50

\* Parameter :data[13] : if the device type is T5A，so:

DATA Add card ;delete card ;stress/duress card; special information Byte 1-4 5-8 9-12 13

The special information define as below:

Byte 0：Add card

Byte 1：Delete card

Byte 2：Stress/duress

If the device type is T50，so:

DATA Password length+ passwod Reserved

Byte 1-3 4-13

Password length = Byte(1) >> 4

Byte[3] padding; //data structure alignment, invalid data

### 2.50.4 Sample

int ret = CChex\_GetAdminCardnumberPassword(anviz\_handle, dev\_idx);

### 2.50.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.51 CChex\_SetAdminCardnumberPassword

### 2.51.1 Description function

【Function】Setup administrator card or password. Only for T5/M5/T50M

Setup T5A/M5A admin card and T50M admin password.

### 2.51.2 Request

【Mode】int CChex\_SetAdminCardnumberPassword(IntPtr CchexHandle, int DevIdx,byte[] Data, int DataLen);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Data[13]：Parameter refer to 2.47 :data[13] description ，Input[Parameter]；

DataLen： length = 13，Input[Parameter]；

### 2.51.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_ADMIN\_CARD\_PWD\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.51.4 Sample

int ret = CChex\_SetAdminCardnumberPassword(CchexHandle,DevIdx,Data,DataLen);

### 2.51.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.52 CChex\_GetDSTParam

### 2.52.1 Description function

【Function】Read the daylight saving time parameters

Read the flag of daylight saving time and time zone .

### 2.52.2Request

【Mode】int CChex\_GetDSTParam(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.52.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_DST\_PARAM\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCCHEX\_SET\_DST\_PARAM\_STRU param; //16byte

byte enabled; //0-Disable 1-enable；

byte ate\_week\_type; //1-Date format 2-week format；

GET\_DST\_PARAM\_TIME start\_time; //7 byte

byte month;

byte day;

byte week\_num; //Weekly definition：0x01-0x04： First 1-4 week 0x81-0x82：Last1-2 week

byte flag\_week; //Week flag: flag\_week 0-6:(Binary 0000000 means：Sat Fri Thu Wed Tue Mon Sun)

byte hour;

byte minute;

byte sec;

GET\_DST\_PARAM\_TIME special\_time； //7 byte

### 2.52.4 Sample

int ret = CChex\_GetDSTParam(anviz\_handle, dev\_idx);

### 2.52.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.53 CChex\_SetDSTParam

### 2.53.1 Description function

【Function】Setup daylight saving time

Setup daylight saving time flag and time zone

### 2.53.2 Request

【Mode】int CChex\_SetDSTParam(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_DST\_PARAM\_STRU Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：16byte，Input[Parameter]；

byte enabled; //0-disable 1-enable；

byte ate\_week\_type; //1-Date format 2-week format；

GET\_DST\_PARAM\_TIME start\_time; //7 byte

byte month;

byte day;

byte week\_num; //Weekly definition：0x01-0x04： First 1-4 week 0x81-0x82：Last1-2 week

byte flag\_week; //Week flag: flag\_week 0-6:(Binary 0000000 means：Sat Fri Thu Wed Tue Mon Sun)

byte hour;

byte minute;

byte sec;

GET\_DST\_PARAM\_TIME special\_time； //7 byte

### 2.53.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_DST\_PARAM\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.53.4 Sample

int ret = CChex\_SetDSTParam(CchexHandle,DevIdx,ref Param);

### 2.53.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.54 CChex\_GetDevExtInfo

### 2.54.1 Description function

【Function】Gets the device extension information code

Read the factory name/Tax number/factory Address

### 2.54.2 Request

【Mode】int CChex\_GetDevExtInfo(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.54.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_DEV\_EXT\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_SET\_DEV\_EXT\_INFO\_STRU param; //320byte

Byte[50] manufacturer\_name; //Factory name

Byte[100] manufacturer\_addr; //Address

Byte[15] duty\_paragraph; //Tax number

Byte[155] reserved; //Reserved

### 2.54.4 Sample

int ret = CChex\_GetDevExtInfo(CchexHandle,DevIdx);

### 2.54.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.55 CChex\_SetDevExtInfo

### 2.55.1 Description function

【Function】Modify the device extension information code

Modify the factory name/Tax number/factory Address

### 2.55.2 Request

【Mode】int CChex\_SetDevExtInfo(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_DEV\_EXT\_INFO\_STRU Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：320byte，Input[Parameter]；

Byte[50] manufacturer\_name; //factory name

Byte[100] manufacturer\_addr; //factory address

Byte[15] duty\_paragraph; //Tax number

Byte[155] reserved; //Reserved

### 2.55.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_DEV\_EXT\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.55.4 Sample

int ret = CChex\_SetDevExtInfo(CchexHandle,DevIdx,ref Param);

### 2.55.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.56 CChex\_GetBasicConfigInfo3

### 2.56.1Description function

【Function】Get the device configuration information3

Read Wiegand mode。

### 2.56.2 Request

【Mode】int CChex\_GetBasicConfigInfo3(IntPtr CchexHandle,int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.56.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_BASIC\_CFG3\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

CCHEX\_SET\_BASIC\_CFG\_INFO3\_STRU param; //15 byte

Byte wiegand\_type; //wiegand mode

Byte online\_mode; //online mode

Byte collect\_level; //Acquisition threshold

Byte pwd\_status; //communication password status =0 ,connect don’t need the communication password by TCP/IP; communication password status =1, send 0x04 command, verify the communication password by TCP/IP

Byte sensor\_status; //=0 don’t report the door sensor status; =1 activate report the door sensor status（the device will activate send the response of 0x2F command )

Byte[8] reserved; //Reserved

Byte independent\_time; //User independent time limit

Byte m5\_t5\_status; //= 0 disable; = 1 enable; Out =2, disable In =4 disable，out =5 disable，In

Byte padding; //alignment invalid data

### 2.56.4 Sample

int ret = CChex\_GetBasicConfigInfo3(CchexHandle,DevIdx);

### 2.56.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.57 CChex\_SetBasicConfigInfo3

### 2.57.1 Description function

【Function】Setup the device configuration information3

Read Wiegand mode

### 2.57.2 Request

【Mode】int CChex\_SetBasicConfigInfo3(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_BASIC\_CFG\_INFO3\_STRU Config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config：15byte，Input[Parameter]；

Byte wiegand\_type; //wiegand mode

Byte online\_mode; //Online mode

Byte collect\_level; //Acquisition threshold

Byte pwd\_status; //communication password status =0 ,connect don’t need the communication password by TCP/IP; communication password status =1, send 0x04 command, verify the communication password by TCP/IP

Byte sensor\_status; //=0 don’t report the door sensor status; =1 activate report the door sensor status（the device will activate send the response of 0x2F command )

Byte[8] reserved; //Reserved

Byte independent\_time; //User independent time limit

Byte m5\_t5\_status; //= 0 disable; = 1 enable; Out =2, disable In =4 disable，out =5 disable，In

### 2.57.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_SET\_BASIC\_CFG3\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.57.4 Sample

int ret = CChex\_SetBasicConfigInfo3(CchexHandle,DevIdx,ref Config);

### 2.57.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.58 CChex\_ConnectionAuthentication

### 2.58.1 Description function

【Function】connection identification

connection identification, identification successful, then response the other command; if the identification successful and no data transfer in 5 mins , recover to dis-identification mode

### 2.58.2 Request

【Mode】int CChex\_ConnectionAuthentication(IntPtr CchexHandle, int DevIdx,ref CCHEX\_CONNECTION\_AUTHENTICATION\_STRU Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：24byte，Input[Parameter]；

Byte[12] username;

Byte[12] password;

//Iris user name “admin”，password “admin”，

//Other devices, don’t identify user name, only identify communication password.

### 2.58.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_CONNECTION\_AUTHENTICATION\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.58.4 Sample

int ret = CChex\_ConnectionAuthentication(CchexHandle,DevIdx,ref Param);

### 2.58.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.59 CChex\_GetRecordNumByEmployeeIdAndTime

### 2.59.1 Description function

【Function】Get the record quantity by user ID and time. Only support A20/972 hardware platform devices.

Get the record quantity by user ID and time.

### 2.59.2 Request

【Mode】int CChex\_GetRecordNumByEmployeeIdAndTime(IntPtr CchexHandle, int DevIdx,ref CCHEX\_GET\_RECORD\_INFO\_BY\_TIME Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：13byte，Input[Parameter]；

Byte[5] EmployeeId; //User ID

Byte[4] start\_date; //The number of seconds after 2000.1.2 Byte[4] end\_date; //The number of seconds after 2000.1.2

User ID = 0xFF means all users

Start date = 0xFF, means unlimit the start date

End date = 0xFF, means unlimit the end date

### 2.59.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_RECORD\_NUMBER\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

int record\_num; // record quantity

### 2.59.4 Sample

int ret = CChex\_GetRecordNumByEmployeeIdAndTime(CchexHandle,DevIdx,ref Param);

### 2.59.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.60 CChex\_DownloadRecordByEmployeeIdAndTime

### 2.60.1 Description function

【Function】Get the record by user ID and time. Only support A20/972 hardware platform devices.

Get the record by user ID and time。

### 2.60.2 Request

【Mode】int CChex\_DownloadRecordByEmployeeIdAndTime(IntPtr CchexHandle, int DevIdx,ref CCHEX\_GET\_RECORD\_INFO\_BY\_TIME Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：13byte，Input[Parameter]；

Byte[12] EmployeeId; //user ID

Byte[4] start\_date; //The number of seconds after 2000.1.2

Byte[4] end\_date; //The number of seconds after 2000.1.2

User ID = 0xFF means all users

Start date = 0xFF, means unlimit the start date

End date = 0xFF, means unlimit the end date

### 2.60.3Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_RECORD\_BY\_EMPLOYEE\_TIME\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

Byte[5] EmployeeId; //User ID

Byte[4] char date; //date time

Byte char back\_id; //back ID

Byte ecord\_type; //record type

Byte[3] work\_type; //work code

Byte[2] padding; //alignment invalid data

Note: The number of times this type of data is returned based on the number of records recorded during the output download time

### 2.60.4 Sample

int ret = CChex\_DownloadRecordByEmployeeIdAndTime(CchexHandle,DevIdx,ref Param);

### 2.60.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.61 Send attendance records in real time

### 2.61.1 Description function

【Function】Send the records in real time

After verification, the attendance information will be output automatically, and only the response information will be available.

### 2.61.2 Request

【Mode】

【Parameter】

### 2.61.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_LIVE\_SEND\_ATTENDANCE\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

byte[5] EmployeeId; //user ID

byte[4] timestamp; //The number of seconds after 2000.1.2

Byte backup; //backup ID

byte record\_type; //record type

byte[3] work\_type[3]; //workcode

byte[2] padding[2];

### 2.61.4Sample

### 2.61.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

2. Set whether the device configuration information 2 is sent in real time

## 2.62 CChex\_GetRecordInfo

### 2.62.1 Description function

【Function】Get the record information

Get the record information, contains already register user quantity, already register FP quantity, already register password quantity, all record quantity and new record quantity

### 2.62.2 Request

【Mode】int CChex\_GetRecordInfoStatus(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.62.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_RECORD\_INFO\_STATUS\_TYPE

Data Part：

uint MachineId; // Device ID

uint EmployeeNum; //User quantity

uint FingerPrtNum; //FP quantity

uint PasswdNum;

uint CardNum;

uint TotalRecNum;

uint NewRecNum;

### 2.62.4 Sample

int ret = CChex\_GetRecordInfoStatus(CchexHandle,DevIdx);

### 2.62.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.63 CChex\_DownloadAllNewRecords

### 2.63.1 Description function

【Function】Download all record

### 2.63.2Request

【Mode】int CChex\_DownloadAllNewRecords(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

### 2.63.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return CCHEX\_RET\_GET\_NEW\_RECORD\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

byte NewRecordFlag; // new record or no

byte[5] EmployeeId; // User ID，

byte[4] Date; // Time，The number of seconds after 2000.1.2

byte BackId; // Backup ID，3 digits：Card，2 digits：，1~0：11，digits 7~4：10 FP templates（1~10）

byte RecordType // Record type，7 digits：1：open the door，0：can’t open the door；3~0 digits attendance status

byte[3] WorkType; // workcode

byte Rsv // reserved

### 2.63.4 Sample

int ret = CChex\_DownloadAllNewRecords(anviz\_handle, dev\_idx);

### 2.63.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.64 Upload user information

【Function】upload user information

### 2.64.2 Request

【Mode】call the universal interface **2.20**  CChex\_ModifyPersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_RET\_PERSON\_INFO\_STRU personlist, byte person\_num);

**Call by the device type：dev->DevTypeFlag & 0xff**

Universal 1：dev->DevTypeFlag & 0xff == 0x01

CChex\_UploadEmployeeInfo(IntPtr \*CchexHandle, int DevIdx, ref CCHEX\_EMPLOYEE\_INFO\_STRU EmployeeList, byte EmployeeNum);

ASCII universal：dev->DevTypeFlag & 0xff == 0x02

CChex\_UploadEmployee2Info(IntPtr \*CchexHandle, int DevIdx, ref CCHEX\_EMPLOYEE2\_INFO\_STRU

EmployeeList, byte EmployeeNum);

UNICODE universal：dev->DevTypeFlag & 0xff == 0x04

CChex\_UploadEmployeeInfo(IntPtr \*CchexHandle, int DevIdx, ref CCHEX\_EMPLOYEE2UNICODE\_INFO\_STRU

EmployeeList, byte EmployeeNum);

UNICODE\_W2 universal：dev->DevTypeFlag & 0xff == 0x20

CChex\_UploadEmployeeInfo(IntPtr \*CchexHandle, int DevIdx, ref CCHEX\_EMPLOYEE2W2\_INFO\_STRU

EmployeeList, byte EmployeeNum);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

EmployeeNum：Increase the number of employees (8 or 12 per upload per device type and customization)

Universal 1：dev->DevTypeFlag & 0xff == 0x01

CCHEX\_EMPLOYEE\_INFO\_STRU //28byte

byte[5] EmployeeId; //user id only one mark

byte[3] Passwd; //password

byte[3] CardId; //card ID

byte[10] EmployeeName; //name

byte DepartmentId; //department ID

byte GroupId; //Group ID

byte Mode; //Attendance mode

byte[2] FpStatus; //FP registration status

byte Special; //Count of the in formations

byte Padding; //Invalid structure alignment

ASCII Universal：dev->DevTypeFlag & 0xff == 0x02

CCHEX\_EMPLOYEE2\_INFO\_STRU //32byte

byte[5] EmployeeId; //user id only one mark

byte[3] Passwd; //Password

byte[4] CardId; //Card Number

byte[10] EmployeeName; //Name

byte DepartmentId; //Department ID

byte GroupId; //Group ID

byte Mode; //Time attendance mode

byte[2] FpStatus; //Fingerprint Status

byte PwdH8bit; //password high 8 bits

byte Rserved; //Save

byte Special; //Count of the in formations

Byte[2] Padding; //Invalid structure alignment

UNICODE Universal：dev->DevTypeFlag & 0xff == 0x04

CCHEX\_EMPLOYEE2UNICODE\_INFO\_STRU //40byte

byte[5] EmployeeId; //user id only one mark

byte[3] Passwd; //Password

byte[4] CardId; //Card Number

byte[20] EmployeeName; //Name

byte DepartmentId; //Department ID

byte GroupId; //Group ID

byte Mode; //Time attendance mode

byte[2] FpStatus; //Fingerprint Status

byte PwdH8bit; //Password high 8 bits

byte Rserved; //Save

byte Special; //Count of the in formations

UNICODE\_W2 Universal：dev->DevTypeFlag & 0xff == 0x20

CCHEX\_EMPLOYEE2W2\_INFO\_STRU

CCHEX\_EMPLOYEE2UNICODE\_INFO\_STRU //40byte

byte[5] EmployeeId; //user id only one mark

byte[3] Passwd; //Password

byte[4] CardId; //Card Number

byte[20] EmployeeName; //Name

byte DepartmentId; //Department ID

byte GroupId; //Group ID

byte Mode; //Time attendance mode

byte[2] FpStatus; //Fingerprint Status

byte PwdH8bit; //Password high 8 bits

byte Rserved; //Save

byte Special; //Count of the in formations

byte[4] start\_date //Employee starting time，The number of seconds after 2000.1.2

byte[4] end\_date //Employee ending time，The number of seconds after 2000.1.2

### 2.64.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update, Type return

Universal 1：dev->DevTypeFlag & 0xff == 0x01

CCHEX\_RET\_ULEMPLOYEE\_INFO\_TYPE

ASCII Universal ：dev->DevTypeFlag & 0xff == 0x02

CCHEX\_RET\_ULEMPLOYEE2\_INFO\_TYPE

UNICODE Universal：dev->DevTypeFlag & 0xff == 0x04

CCHEX\_RET\_ULEMPLOYEE2UNICODE\_INFO\_TYPE

UNICODE\_W2 Universal：dev->DevTypeFlag & 0xff == 0x20

CCHEX\_RET\_ULEMPLOYEE2W2\_INFO\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:OK, -1:error

### 2.64.4 Sample

int ret = CChex\_UploadEmployeeInfo(anviz\_handle, dev\_idx，EmployeeList，1);

### 2.64.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.；

## 2.65 Chex\_GetBasicConfigInfo5

### 2.65.1 Function

【Function】获取考勤机配置信息5（Bolid定制）

### 2.65.2 Request

【Mode】int CChex\_GetBasicConfigInfo5(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；参；

### 2.65.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_GET\_BASIC\_CFG5\_TYPE

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

byte fail\_alarm\_time; // 勤失败报警次数: 0: 不报警1~10 N次失败报警

byte tamper\_alarm; // 防拆报警: 0: 关闭1: 开启

byte[94] reserved; // reserved

### 2.65.4 Sample

int ret = CChex\_GetBasicConfigInfo5(anviz\_handle, dev\_idx);

### 2.65.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.66 CChex\_SetBasicConfigInfo 5

### 2.66.1 Function

【Function】Setup Time attendance device information 5（Bolid Customize）

### 2.66.2 Request

【Mode】int CChex\_SetBasicConfigInfo5(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_BASIC\_CFG\_INFO5\_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param: //96 byte （0xFF means do not setup）

byte fail\_alarm\_time; // Times of Attendance Fail Alarm: 0: Do not alarm out 1~10 times of alarm out

byte tamper\_alarm; // Tamper alarm: 0: Close1: Active

byte[94] reserved; // reserved

### 2.66.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_SET\_BASIC\_CFG5\_TYPE

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

### 2.66.4 Sample

int ret = CChex\_SetBasicConfigInfo5(anviz\_handle, dev\_idx,ref Param);

### 2.66.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

Function

## 2.67 CChex\_GetCardNo

### 2.67.1 Function

【Function】Query T5S swipe card information（Bolid Customize）

**Notice**： 1：The first time call the function ：Setting the device as swipe card mode

2：Swipe card over time > 5sec；

3：The second time call the function ：Return the last Card number before called the function.

### 2.67.2 Request

【Mode】int Chex\_GetCardNo(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.67.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_GET\_CARD\_ID\_TYPE

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

Uint card\_id; //Card Number

### 2.67.4 Sample

int ret = Chex\_GetCardNo(anviz\_handle, dev\_idx);

### 2.67.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.68 CChex\_SetDevCurrentStatus

### 2.68.1 Function

【Function】User to modify the device status by temporary（Bolid Customize）

### 2.68.2 Request

【Mode】int CChex\_SetDevCurrentStatus(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_DEV\_CURRENT\_STATUS\_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param: //96 byte （0xFF means do not setup）

byte alarm\_stop; //Disable the Alarm except 0XFF

byte door\_status //0: Setting the access control as normal 1: Setting the access control as normal Open 2: Setting the access control as normal Close

byte[94] reserved; // reserved

### 2.68.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_SET\_DEV\_CURRENT\_STATUS\_TYPE

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

### 2.68.4 Sample

int ret = CChex\_SetDevCurrentStatus(anviz\_handle, dev\_idx,ref Param);

### 2.68.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.69 CChex\_GetServiceURL

### 2.69.1 Function

【Function】Get the server’s URL

### 2.69.2 Request

【Mode】int CChex\_GetServiceURL(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

### 2.69.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update return CCHEX\_RET\_GET\_URL\_TYPE

Data Part：

uint MachineId; // device ID

Int Result; // 0:Success -1：Fail

Byte[4] Dns //dns address

byte[100] Url; //URL address

### 2.69.4 Sample

int ret = CChex\_GetServiceURL(anviz\_handle, dev\_idx);

### 2.69.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.70 CChex\_SetServiceURL

### 2.70.1 Function

【Function】Setuphe server’s URL

### 2.70.2 Request

【Mode】int CChex\_SetServiceURL(IntPtr CchexHandle, int DevIdx,ref CCHEX\_SET\_URL\_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

Param: //104 byte

Byte[4] Dns //dns Address

byte[100] Url; //URL Address

### 2.70.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type return CCHEX\_RET\_SET\_URL\_TYPE

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Success -1：Fail

### 2.70.4 Sample

int ret = CChex\_SetServiceURL(anviz\_handle, dev\_idx,ref Param);

### 2.70.15 Notice

1.Make sure the CChex\_Start already active before running

## 2.71 CChex\_UploadFile

### 2.71.1 Function

【Function】Update firmware, Image and Voice

### 2.71.2 Request

【Mode】int CChex\_UploadFile(IntPtr CchexHandle, int DevIdx, byte FileType, byte[] FileName, byte[] Buff, int Len);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

FileType: = 0 Firmware，= 1 Image，= 2 Voice，= 3 Language

FileName: The name update to the device（MAX\_len = 10)；

Buff: The uploaded file is read as a string;

Len： The length of the string of the uploaded file.

### 2.71.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type return CCHEX\_RET\_UPLOADFILE\_TYPE

Data Part：

uint MachineId; // Device ID

int Result; // 0:Success -1：Fail

uint TotalBytes; // Total length of the uploaded string

uint SendBytes; // Length already uploaded

**Notice：Every 32K of data is uploaded will return the type once SendBytes == TotalBytes，the upload is completed.**

### 2.71.4 Sample

int ret = CChex\_UploadFile(CchexHandle,DevIdx,FileType,FileName,Buff,Len);

### 2.71.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.72 CChex\_UpdateDevStatus

### 2.72.1 Function

【Function】ask device the status of the firmware updating

**Notice:** After the upgrade package is uploaded, it is called multiple times. When the result is: 1: Check completion 2: The verification is successful. You can restart the upgrade

### 2.72.2 Request

【Mode】int CChex\_UpdateDevStatus(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

### 2.72.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type return CCHEX\_RET\_UPDATEFILE\_STATUS\_TYPE

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Success -1：Fail

int verify\_status; // 0:verifying ;1: Verify Success

int verify\_ret; // 0:Verify Success;1: Verify Fail

### 2.72.4 Sample

int ret = CChex\_UpdateDevStatus(anviz\_handle, dev\_idx);

### 2.72.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.73 CChex\_GetStatusSwitch

### 2.73.1 Function

【Function】Get status switch information, total with 16 status

**Notice：Device TypeDevTypeFlag & 0x200000 = = 1 can be called**

### 2.73.2 Request

【Mode】int CChex\_GetStatusSwitch(IntPtr CchexHandle, int DevIdx,byte GroupId);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

GroupId：Edit Group ID

### 2.73.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_GET\_STATUS\_SWITCH\_TYPE

Data Part：

uint MachineId; // Device IDInt Result; // 0:Success -1：Fail

byte group\_id; //get the group number

CCHEX\_GET\_PERIOD\_TIME\_ONE\_STRU\_EXT\_INF day\_week[7]; //7 groups as 7day for one week. One group with 28Byte

byte StartHour; //Starting Hour

byte StartMin; //Starting Mins

byte EndHour; //Ending Hour

byte EndMin; //Ending Mins

byte status\_id; //Setup status number

Byte[2] padding; //Invalid data, fill in the structure

### 2.73.4 Sample

int ret = CChex\_GetStatusSwitch(anviz\_handle, dev\_idx,GroupId);

### 2.73.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.74 CChex\_SetStatusSwitch

### 2.74.1 Function

【Function】Setup status switch information, total with 16 status

**Notice：Device Type DevTypeFlag & 0x200000 = = 1 can be called**

### 2.74.2 Request

【Mode】int CChex\_SetStatusSwitch(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_STATUS\_SWITCH\_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

Param： // 32 byte

byte group\_id; //get the group number

CCHEX\_GET\_PERIOD\_TIME\_ONE\_STRU\_EXT\_INF day\_week[7]; // 7 groups as 7day for one week. One group with 28Byte

byte StartHour; //Starting Hour

byte StartMin; //Starting Mins

byte EndHour; //Ending Hour

byte EndMin; //Ending Mins

byte status\_id; //Setup status number

Byte[2] padding; //Invalid data, fill in the structure

### 2.74.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_SET\_STATUS\_SWITCH\_TYPE

Data Part：

uint MachineId; // Device IDInt Result; // 0:Success -1：Fail

### 2.74.4 Sample

int ret = CChex\_SetStatusSwitch(anviz\_handle, dev\_idx,ref Param);

### 2.74.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.75 CChex\_GetStatusSwitch\_EXT

### 2.75.1 Function

【Function】Get status switch information

**Notice：Device TypeDev TypeFlag & 0x100000 = = 1 can be called**

### 2.75.2 Request

【Mode】int CChex\_GetStatusSwitch\_ext(IntPtr CchexHandle, int DevIdx,byte FlagWeek);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

FlagWeek：Week day as 0-6，to represent Sun. to Sat., Such as Monday ：00000010

### 2.75.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type return CCHEX\_RET\_GET\_STATUS\_SWITCH\_EXT\_TYPE

Data Part：

uint MachineId; // Device IDInt Result; // 0:Success -1：Fail

byte flag\_week; //get the group number

CCHEX\_ONE\_TIMER\_STATUS one\_time[8]; //8 time zone 40Byte

byte StartHour; //Starting Hour

byte StartMin; //Starting Mins

byte EndHour; //Ending Hour

byte EndMin; //Ending Mins

byte status\_id; //Setup status number

Byte[3] padding; //Invalid data, fill in the structure

### 2.75.4 Sample

int ret = CChex\_GetStatusSwitch\_EXT(anviz\_handle, dev\_idx,flag\_week);

### 2.75.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.76 CChex\_SetStatusSwitch\_EXT

### 2.76.1 Function

【Function】Setup time attendance status information

**Notic：Device type DevTypeFlag & 0x100000 = = 1 can be call**

### 2.76.2 Request

【Mode】int CChex\_SetStatusSwitch\_EXT(IntPtr CchexHandle, int DevIdx, ref CCHEX\_SET\_STATUS\_SWITCH\_STRU\_EXT Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

Param： // 44 byte

byte flag\_week; //get the group number

CCHEX\_ONE\_TIMER\_STATUS one\_time[8]; //8 timezone 40Byte

byte StartHour; //Starting Hour

byte StartMin; //Starting Mins

byte EndHour; //Ending Hour

byte EndMin; //Ending Mins

byte status\_id; //Setup status number

Byte[3] padding; //Invalid data, fill in the structure

### 2.76.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex\_Update Type return CCHEX\_RET\_SET\_STATUS\_SWITCH\_EXT\_TYPE

Data Part：

uint MachineId; // Device IDInt Result; // 0:Success -1：Fail

### 2.76.4 Sample

int ret = CChex\_SetStatusSwitch\_EXT(anviz\_handle, dev\_idx,ref Param);

### 2.76.5 Notice

1. Make sure the CChex\_Start already active before running

## 2.77 CChex\_Get\_Service\_Port

### 2.77.1 Function

【Function】Setup the current server port

### 2.77.2 Request

【Mode】int CChex\_Get\_Service\_Port(IntPtr CchexHandle);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

### 2.77.3 Sample

int ret = CChex\_Get\_Service\_Port(CchexHandle);

【Return value】Returnret as server port

## 2.78 CChex\_SetSdkConfig

### 2.78.1 Function

【Function】Monitor the records status, the new record will automatic download and after download new records automatic to make a flag and SDK will print out to the LOG folder.

\* after CChex\_Start();

\* para :

\* SetRecordflag = 1 set recordflag after download new record;else = 0

\* SetLogFile = 1 set some info log to file for find problem ;else = 0

\* if do not set "CChex\_SetSdkConfig(void \*CchexHandle,int SetAutoDownload,int SetRecordflag,int SetLogFile)", config is default

\* ANVIZ\_DEFAULT:

\* W2 : SetRecordflag = 1,SetLogFile = 0,SetAutodownload = 1;

\* SEATS : SetRecordflag = 1,SetLogFile = 0,SetAutodownload = 1;

\* DR : SetRecordflag = 0,SetLogFile = 0,SetAutodownload = 1;

\* COMMON : SetRecordflag = 0,SetLogFile = 0,SetAutodownload = 1;

\* Bolid : SetRecordflag = 1,SetLogFile = 0,SetAutodownload = 1;

### 2.78.2 Request

【Mode】void CChex\_SetSdkConfig(IntPtr CchexHandle, int SetAutoDownload,int SetRecordflag, int SetLogFile);

【Parameter】SetAutodownload: = 1: Monitor records status , when get new records automatic download record,others do not download automatic

SetRecordflag: = 1:After download new records automatic to make a flag.

SetLogFile: = 1:SDK Print out to Log file

### 2.78.3 Sample

int ret = CChex\_SetSdkConfig(CchexHandle,1,1,0);

;

## 2.79 CChex\_UploadRecord

### 2.79.1 Function

【Function】Upload Time Attendance Records

### 2.79.2 Request

【Mode】int CChex\_UploadRecord(IntPtr CchexHandle, int DevIdx, ref CCHEX\_UPLOAD\_RECORD\_INFO\_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param： //

Byte[5] EmployeeId; //Employee Id

Byte[4] char date; //Date

Byte char back\_id; //Backup ID

Byte ecord\_type; //Record Type

Byte[3] work\_type; //Work Code

### 2.79.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_UPLOAD\_RECORD\_TYPE //87

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

Byte[5] EmployeeId; // digital of Employee Id

Byte[3] padding; // Invalid data, fill in the structure

### 2.79.4 Sample

int ret = CChex\_UploadRecord(anviz\_handle, dev\_idx,ref Param);

### 2.79.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.80 CChex\_UploadRecord\_VER\_4\_NEWID

### 2.80.1 Function

【Function】Upload Time Attendance Records

**(Version: Device Type DevTypeFlag & 0XFF ==DEV\_TYPE\_VER\_4\_NEWID)**

### 2.80.2 Request

【Mode】int CChex\_UploadRecord\_VER\_4\_NEWID(IntPtr CchexHandle, int DevIdx, ref CCHEX\_UPLOAD\_RECORD\_INFO\_STRU\_VER\_4\_NEWID Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param： //

Byte[28] EmployeeId; // String of Employee Id

Byte[4] date; //Date

Byte back\_id; //Backup ID

Byte ecord\_type; //Record Type

Byte[3] work\_type; //Work Code

### 2.80.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_UPLOAD\_RECORD\_TYPE //87

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

Byte[28] EmployeeId; // String of Employee Id

### 2.80.4 Sample

int ret = CChex\_UploadRecord\_VER\_4\_NEWID(anviz\_handle, dev\_idx,ref Param);

### 2.80.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.81 CChex\_GetOnePersonInfo

**(Version :digital of Employee Id )**

## CChex\_GetOnePersonInfo\_VER\_4\_NEWID

**(Version:**

**String of Employee Id DevTypeFlag & 0XFF ==DEV\_TYPE\_VER\_4\_NEWID)**

### 2.81.1 Function

【Function】According to employee Id to get user information

### 2.81.2 Request

【Mode】int CChex\_GetOnePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX\_GET\_ONE\_EMPLOYEE\_INFO\_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param： //

Byte[5] EmployeeId; //digital of Employee Id

【Mode】int CChex\_GetOnePersonInfo\_VER\_4\_NEWID(IntPtr CchexHandle, int DevIdx, ref CCHEX\_GET\_ONE\_EMPLOYEE\_INFO\_STRU\_VER\_4\_NEWID Param);

**(Version :Device Version DevTypeFlag & 0XFF ==DEV\_TYPE\_VER\_4\_NEWID)**

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param： //

Byte[28] EmployeeId; //String of Employee Id

### 2.81.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_GET\_ONE\_EMPLOYEE\_INFO\_TYPE //88

Data Part：

**digital of Employee Id ,Without employee valid time:**

uint MachineId; // Device ID

int CurIdx; // Current index ID

int TotalCnt; // Total number of User

byte [5]EmployeeId; // User ID 5 bytes

byte password\_len; // Length of the Password

byte max\_password; // Max length of the Password, the value is 6 Can not modify

int password; // Password

byte max\_card\_id; // Max length of the Card Number, the value is 6(3 byte) or 10(3 byte) Can not modify

uint card\_id; // Card Number

byte max\_EmployeeName; // Max length of the User Name, the value is 10,20,64 or 64(3 byte) When the value is 160 the user name store in the EmployeeName2

byte[64] EmployeeName; // User Name

byte DepartmentId; // Department ID

byte GroupId; // Group ID

byte Mode; // Time Attendance Mode

uint Fp\_Status; // The status of FP enroll，0~9:fp; 10:face; 11:iris1; 12:iris2

byte Rserved1; // for 22

byte Rserved2; // for 72 and 22

byte Special; // Special information

// DR info

byte[160] EmployeeName2; // User Name2

byte[13] RFC; // RFC Information

byte[18] CURP; // CURP Information

**digital of Employee Id ,With employee valid time:**

uint MachineId; // Device ID

int CurIdx; // Current index ID

int TotalCnt; // Total number of User

byte [5]EmployeeId; // User ID 5 bytes

byte password\_len; // Length of the Password

byte max\_password; // Max length of the Password, the value is 6 Can not modify

int password; // Password

byte max\_card\_id; // Max length of the Card Number, the value is 6(3 byte) or 10(3 byte) Can not modify

uint card\_id; // Card Number

byte max\_EmployeeName; // Max length of the User Name, the value is 10,20,64 or 64(3 byte) When the value is 160 the user name store in the EmployeeName2

byte[64] EmployeeName; // User Name

byte DepartmentId; // Department ID

byte GroupId; // Group ID

byte Mode; // Time Attendance Mode

uint Fp\_Status; // The status of FP enroll，0~9:fp; 10:face; 11:iris1; 12:iris2

byte Rserved1; // for 22

byte Rserved2; // for 72 and 22

byte Special; // Special information

// DR info

byte[160] EmployeeName2; // User Name2

byte[13] RFC; // RFC Information

byte[18] CURP; // CURP Information

byte[4] start\_date; // RFC Information

byte[4] end\_date; // CURP Information

**String of Employee Id , with employee valid time:**

uint MachineId; // Device ID

int CurIdx; // Current index ID

int TotalCnt; // Total number of User

byte [28]EmployeeId; // Enployee ID,28bytes

byte password\_len; // Length of the Password

byte max\_password; // Max length of the Password, the value is 6 Can not modify

int password; // Password

byte max\_card\_id; // Max length of the Card Number, the value is 6(3 byte) or 10(3 byte) Can not modify

uint card\_id; // Card Number

byte max\_EmployeeName; // Max length of the User Name, the value is 10,20,64 or 64(3 byte) When the value is 160 the user name store in the EmployeeName2

byte[64] EmployeeName; // User Name

byte DepartmentId; // Department ID

byte GroupId; // Group ID

byte Mode; // Time Attendance Mode

uint Fp\_Status; // The status of FP enroll，0~9:fp; 10:face; 11:iris1; 12:iris2

byte Rserved1; // for 22

byte Rserved2; // for 72 and 22

byte Special; // Special information

// DR info

byte[160] EmployeeName2; // User Name2

byte[13] RFC; // RFC Information

byte[18] CURP; // CURP Information

byte[4] start\_date; // RFC Information

byte[4] end\_date; // CURP Information

### 2.81.4 Sample

int ret = CChex\_UploadRecord(anviz\_handle, dev\_idx,ref Param);

int ret = CChex\_UploadRecord\_VER\_4\_NEWID(anviz\_handle, dev\_idx,ref Param);

### 2.81.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.82 CChex\_GetMachineId

### 2.82.1 Function

【Function】Read the device communication device ID

### 2.82.2 Request

【Mode】int CChex\_GetMachineId(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

### 2.82.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_GET\_MACHINE\_ID\_TYPE //84

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

uint cur\_machineid; //Communicatrion device ID

### 2.82.4 Sample

int ret = CChex\_GetMachineId(anviz\_handle, dev\_idx);

### 2.82.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.83 CChex\_SetMachineId

### 2.83.1 Function

【Function】Setting device Communication device ID

### 2.83.2 Request

【Mode】int CChex\_SetMachineId(IntPtr CchexHandle, int DevIdx,uint MachineId);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

MachineId: device Communication device ID;

### 2.83.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_SET\_MACHINE\_ID\_TYPE //85

Data Part：

uint MachineId; // Device ID

Int Result; // 0:Succes -1：Failure

uint cur\_machineid; //New Communicatrion device ID

uint old\_machineid; //Old Communicatrion device ID

### 2.83.4 Sample

int ret = CChex\_SetMachineId(anviz\_handle, dev\_idx,MachineId);

### 2.83.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

## 2.84 CChex\_ManageLogRecord

### 2.84.1 Function

【Function】Setting device Communication device ID

### 2.84.2 Request

【Mode】int CChex\_ManageLogRecord(IntPtr CchexHandle, int DevIdx, ref CCHEX\_MANAGE\_LOG\_RECORD Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param： //

Byte[4] start\_date; // The second since of after 2000 years, 2000.1.2

Byte[4] end\_date; //The second since of after 2000 years, 2000.1.2

Byte CmdType; //Extended command ：　0x00 Get the total number of current time periods

0x01 Download the current time period log record

0x02 delete all records in the current time period

0x03 Set real-time report log record

0x04 Get the status of real-time reporting records

Byte AutoFlag; //0:Close real-time report log record 1: Enable real-time report log record

### 2.84.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex\_Update Type Return CCHEX\_RET\_MANAGE\_LOG\_RECORD\_TYPE //90

Data Part：

uint MachineId; // Device ID

uint CmdType; // Extended command

0x00 :Result judge success,TotalNum is Total number of the log

0x02 :Result is valid Delete success or not

0x03 Set real-time report log record

0x04 Get the status of real-time reporting records

0x05 Real-time reporting record

Int Result; // 0:Succes -1：Failure

uint IsAuto; // Auto-tagging

uint TotalNum; // Total number of the log

CmdType == 0x00:Result==0 Total number of the log

CmdType == 0x01:Result==0 Total download number of the log

uint CurNum; // The log id of the current logByte[5] EmployeeId; //Employee Id

Byte[4] Date; //Event date The second since of after 2000 years, 2000.1.2

Byte[2] LogType; //Log Type 0x0001 open door 0x0002 close door

0x0003 door sensor 0x0004 Tamper alarm

0x0005 Exit button 0x0006 burst open the door

Byte[2] LogLen; //Log content length

Byte[3] padding; //Fill alignment invalid data

### 2.84.4 Sample

int ret = CChex\_ManageLogRecord(anviz\_handle, dev\_idx,ref Param);

### 2.84.5 Notice

1. Make sure that CChex\_Start has been started successfully before run.

# Face Deep Series Temperature detection and face template interface

## CChex\_TM\_DownloadAllRecords

### Function Description

【Function】Download the all of user records with temperature detection and mask detection result

### Request

【Mode】int CChex\_TM\_DownloadAllRecords(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_TM\_ALL\_RECORD\_INFO\_TYPE //150

Data Part：CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

[StructLayout(LayoutKind.Sequential, Size = 34, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

        {

            public uint MachineId;         //Device ID

            int Result;                     //0 ok, -1 err

            public byte NewRecordFlag;      //New records or not

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;       //Employee ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[] Date;             //Date & Time

            public byte BackId;             //Backup ID

            public byte RecordType;         //Record Type

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 3)]

            public byte[] WorkType;         //Word Code(ONLY use 3bytes )

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] Temperature;      //Temperature integer value  For example 368 means temperature is 36.8°C

            public byte IsMask;             //Mask Wearing or not

            public byte OpenType;           // Relay work mode

            public uint CurIdx;             //add  VER 22

            public uint TotalCnt;           //add  VER 22

        }

### Sample

int ret =CChex\_TM\_DownloadAllRecords(anviz\_handle, dev\_idx);

### Notice

Please make sure the CChex\_Start has been started successfully before running.

## CChex\_TM\_DownloadAllNewRecords

### Function Description

【Function 】Download new records with the temperature detection and mask detection

### Request

【Mode】int CChex\_TM\_DownloadAllNewRecords(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】 CCHEX\_RET\_TM\_NEW\_RECORD\_INFO\_TYPE //151

Data Part：CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

[StructLayout(LayoutKind.Sequential, Size = 34, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

        {

            public uint MachineId;         //Device ID

            int Result;                     //0 ok, -1 err

            public byte NewRecordFlag;      //New records or not

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;       //Employee ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[] Date;             //Date & Time

            public byte BackId;             //Backup ID

            public byte RecordType;         //Record Type

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 3)]

            public byte[] WorkType;         //Word Code(ONLY use 3bytes )

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] Temperature;      //Temperature integer value     For example 368 means temperature is 36.8°C

            public byte IsMask;             //Mask Wearing or not

            public byte OpenType;           // Relay work mode

            public uint CurIdx;             //add  VER 22

            public uint TotalCnt;           //add  VER 22

        }

### Sample

int ret =CChex\_TM\_DownloadAllNewRecords(anviz\_handle, dev\_idx);

### Notice

Please make sure the CChex\_Start has been started successfully before running.

## CChex\_TM\_UploadRecord

### Function Description

【Function 】Upload the temperature record and mask detection records

Request

【Mode】CChex\_TM\_UploadRecord(IntPtr CchexHandle, int DevIdx, ref CCHEX\_TM\_UPLOAD\_RECORD\_INFO\_STRU Record);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

Record: CCHEX\_TM\_UPLOAD\_RECORD\_INFO\_STRU结构

[StructLayout(LayoutKind.Sequential, Size = 18, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_TM\_UPLOAD\_RECORD\_INFO\_STRU

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;    //Employee ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[] date;          //Date & Time

            public byte back\_id;           //Backup ID

            public byte record\_type;       //Record Type

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 3)]

            public byte[] work\_type;      //Word Code(ONLY use 3bytes )

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] Temperature;      //Temperature integer value       (ONLY use 3bytes )

            public byte IsMask; //Mask Wearing or not

            public byte OpenType; // Relay work mode

        }

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_TM\_UPLOAD\_RECORD\_INFO\_TYPE //153

Data Part：CCHEX\_RET\_TM\_UPLOAD\_RECORD\_STRU (CCHEX\_RET\_DEL\_EMPLOYEE\_INFO\_STRU)

 [StructLayout(LayoutKind.Sequential, Size = 16, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_DEL\_EMPLOYEE\_INFO\_STRU

        {

            public uint MachineId;      //Device ID

            public int Result;          //0 ok, -1 err

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;   //Employee ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 3)]

            public byte[] padding;      //Padded format alignment

        }

### Sample

int ret =CChex\_TM\_UploadRecord(CchexHandle, DevIdx, ref Record);;

### Notice

1. Please make sure the CChex\_Start has been started successfully before running.

## CChex\_TM\_DownloadRecordByEmployeeIdAndTime

### Function Description

【Function 】Download the employee’s temperature and mask detection record by assigned employee ID and time range.

### Request

【Mode】CChex\_TM\_DownloadRecordByEmployeeIdAndTime(IntPtr CchexHandle, int DevIdx, ref CCHEX\_GET\_RECORD\_INFO\_BY\_TIME Param);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

Record: CCHEX\_GET\_RECORD\_INFO\_BY\_TIME

[StructLayout(LayoutKind.Sequential, Size = 13, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_GET\_RECORD\_INFO\_BY\_TIME

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[]   EmployeeId;     //Employee ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[]   start\_date;     //Start time The number of seconds after from 2000.1.2

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[]   end\_date;       //End time The number of seconds after from 2000.1.2

        }

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_TM\_RECORD\_BY\_EMPLOYEE\_TIME\_TYPE //154

Data Part：CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

[StructLayout(LayoutKind.Sequential, Size = 34, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_TM\_RECORD\_INFO\_STRU

        {

            public uint MachineId;         //Device ID

            int Result;                     //0 ok, -1 err

            public byte NewRecordFlag;      //New records or not

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;       //Employee ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[] Date;             //Date & Time

            public byte BackId;             //Backup ID

            public byte RecordType;         //Record Type

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 3)]

            public byte[] WorkType;         //Word Code(ONLY use 3bytes )

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] Temperature;      //Temperature integer value   For example 368 means temperature is 36.8°C

            public byte IsMask;             //Mask Wearing or not

            public byte OpenType;           // Relay work mode

            public uint CurIdx;             //add  VER 22

            public uint TotalCnt;           //add  VER 22

        }

### Sample

int ret =CChex\_TM\_DownloadRecordByEmployeeIdAndTime(CchexHandle, DevIdx, ref Param);;

### Notice

1. Please make sure the CChex\_Start has been started successfully before running.

## CChex\_GetTRecordNumberByType

### Function Description

【Function 】Get the temperature record count

### Request

【Mode】CChex\_GetTRecordNumberByType(IntPtr CchexHandle, int DevIdx, int RecordTpye);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

RecordTpye: Temperature records type 0: All Type 10: Normal 20: Abnormal

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_GET\_T\_RECORD\_NUMBER\_TYPE //155

Data Part：CCHEX\_RET\_GET\_T\_RECORD\_NUMBER\_STRU

[StructLayout(LayoutKind.Sequential, Size = 16, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_GET\_T\_RECORD\_NUMBER\_STRU

        {

            public uint MachineId;         //Device ID

            public int Result; //0 ok, -1 err

            public int RecordNum;          //Record Count

            public int RecordTpye;         //Temperature records type

        }

### Sample

int ret =CChex\_GetTRecordNumberByType(CchexHandle,DevIdx, 10);

### Notice

1. Please make sure the CChex\_Start has been started successfully before running.

## CChex\_GetTRecordByType

### Function Description

【Function 】According to the temperature type to download the record (Temperature type: Normal, Abnormal)

### Request

【Mode】CChex\_GetTRecordByType(IntPtr CchexHandle, int DevIdx, int RecordTpye);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device,Input [Parameter]

RecordTpye: Temperature records type 0: All Type 10: Normal 20: Abnormal

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_GET\_T\_RECORD\_TYPE//156

Data Part：CCHEX\_RET\_GET\_T\_RECORD\_STRU

[StructLayout(LayoutKind.Sequential, Size = 32, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_GET\_T\_RECORD\_STRU

        {

            public uint MachineId;         //Device ID

            public int Result; //0 ok, -1 err

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 8)]

            public byte[] RecoradId;       //Temperature Detection Record ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 4)]

            public byte[] Date;             //Date & Time The number of seconds after from 2000.1.2

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] Temperature;      // Temperature integer value divide 10, Temperature integer/10  368/10=36.8

            public byte TemperatureType;    //10-Normal, 20-Abnormal

            public byte MaskType;           //Mask Wearing or not/ 0:do not wearing mask，1:Wearing Mask， 2 The FaceDeep hardware do not detection the mask.

            public int CurIdx;              //Current running number

            public int TotalCnt;            //Current download count        }

### Sample

int ret =CChex\_GetTRecordByType(CchexHandle,DevIdx, 10);

### Notice

1. Please make sure the CChex\_Start has been started successfully before running.

## CChex\_GetPictureByTRecordIdType

### Function Description

【Function 】Get the photo by assigned temperature detection record ID Request

【Mode】CChex\_GetPictureByTRecordIdType(IntPtr CchexHandle, int DevIdx, ref CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU Param);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

Param: CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU

[StructLayout(LayoutKind.Sequential, Size = 9, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 8)]

            public byte[] RecoradId;        //Temperature Detection Record ID

            public byte TemperatureType;    //0-All Type 10-Normal ,20- Abnormal

        }

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_GET\_T\_PICTURE\_BY\_RECORD\_ID\_TYPE //157

Data Part：CCHEX\_RET\_GET\_PICTURE\_BY\_RECORD\_ID\_STRU

[StructLayout(LayoutKind.Sequential, Size = 24, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_GET\_PICTURE\_BY\_RECORD\_ID\_STRU

        {

            public uint MachineId;              //Device ID

            public int Result;                  //0:Success -1:Failed

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 8)]

            public byte[] RecoradId;    //Temperature Detection Record ID

            public byte TemperatureType;  //0-All Type 10-Normal ,20- Abnormal

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 3)]

            public byte[] Padding;       //Invalid Data to fill in the structure

            public uint DataLen;                //The length of the photo

            //[MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 0)]

            //public byte[] PictureData;        //Photo content

        }

### Sample

int ret =CChex\_GetPictureByTRecordIdType(CchexHandle, DevIdx, ref Param);

### Notice

1. Please make sure the CChex\_Start has been started successfully before running.

## CChex\_DelPictureByTRecordIdType

### Function Description

【Function】Delete the assigned Photo by temperature detection record IDRequest

【Mode】CChex\_DelPictureByTRecordIdType(IntPtr CchexHandle, int DevIdx, ref CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU Param);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

Param: CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU

[StructLayout(LayoutKind.Sequential, Size = 9, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_PICTURE\_BY\_RECORD\_ID\_STRU

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 8)]

            public byte[] RecoradId;        //Temperature detection ID

            public byte TemperatureType;   //0-All Type 10-Normal ,20- Abnormal

        }

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_DEL\_T\_PICTURE\_BY\_RECORD\_ID\_TYPE //158

Data Part：CCHEX\_RET\_DEL\_PICTURE\_BY\_RECORD\_ID\_STRU

[StructLayout(LayoutKind.Sequential, Size = 17, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_RET\_DEL\_PICTURE\_BY\_RECORD\_ID\_STRU

        {

            public uint MachineId;                  //Device ID

            public int Result;                      //0 ok, -1 err

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 8)]

            public byte[] RecoradId;                //Temperature detection ID

            public byte TemperatureType;   //0-All Type 10-Normal ,20- Abnormal

        }

### Sample

int ret =CChex\_DelPictureByTRecordIdType(CchexHandle,DevIdx, ref Param);

### Notice

1. Please make sure the CChex\_Start has been started successfully before running.

## CChex\_DownloadFacePictureModule

### Function Description

【Function 】Face Template download (Photo Format)

### Request

【Mode】 CChex\_DownloadFacePictureModule(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_PERSON\_INFO\_STRU Param);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device,Input [Parameter]

Param: CCHEX\_FACE\_PICTURE\_MODULE\_STRU (CCHEX\_DEL\_PERSON\_INFO\_STRU)

[StructLayout(LayoutKind.Sequential, Size = 6, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_DEL\_PERSON\_INFO\_STRU

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;  // Employee ID

            public byte operation;     //The location number of the backup temperature. Fingerprint 1-10. The face and Iris template do not need setup the parameter

        }

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_DOWNLOAD\_FACE\_PICTURE\_MODULE\_TYPE //159

Data Part：CCHEX\_DOWNLOAD\_FACE\_PICTURE\_MODULE

[StructLayout(LayoutKind.Sequential, Size = 20, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_DOWNLOAD\_FACE\_PICTURE\_MODULE

        {

            public uint MachineId;

            public int Result;

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;                     //Employee ID

            public byte Backup;                       //Backup ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] padding;                        //Invalid Data to fill in the structure

            public uint DataLen;                          //The length of the face template data

            //[MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 0)]

            //public byte[] PictureData;                                        //face template data

        }

### Sample

int ret =CChex\_DownloadFacePictureModule(CchexHandle, DevIdx, ref Param);

### Notice

Please make sure the CChex\_Start has been started successfully before running.

## CChex\_UploadFacePictureModule

### Function Description

【Function 】upload the face template

Request

【Mode】 CChex\_UploadFacePictureModule(IntPtr CchexHandle, int DevIdx, ref CCHEX\_DEL\_PERSON\_INFO\_STRU Param, byte[] PictureBuff, int BuffLen);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

Param: CCHEX\_FACE\_PICTURE\_MODULE\_STRU 结构 (CCHEX\_DEL\_PERSON\_INFO\_STRU)

[StructLayout(LayoutKind.Sequential, Size = 6, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_DEL\_PERSON\_INFO\_STRU

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;  // Employee ID

            public byte operation;     //The location number of the backup temperature. Fingerprint 1-10. The face and Iris template do not need setup the parameter

        }

PictureBuff: The data of the face template;

BuffLen : The length of the data;

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_UPLOAD\_FACE\_PICTURE\_MODULE\_TYPE //160

Data Part：CCHEX\_UPLOAD\_FACE\_PICTURE\_MODULE

[StructLayout(LayoutKind.Sequential, Size = 16, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_UPLOAD\_FACE\_PICTURE\_MODULE

        {

            public uint MachineId;

            public int Result;

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;                                       //Employee ID

            public byte Backup;                         //Backup ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] padding;      //Invalid Data to fill in the structure

        }

### Sample

int ret =CChex\_UploadFacePictureModule(CchexHandle,DevIdx, ref Param, bPictureBuff, BuffLen);

### Notice

Please make sure the CChex\_Start has been started successfully before running.

## CCHex\_AddFingerprintOnline\_FacePicture

### Function Description

【Function 】Remote enroll user face

### Request

【Mode】 CCHex\_AddFingerprintOnline\_FacePicture(IntPtr CchexHandle, int DevIdx, ref CCHEX\_ADD\_FINGERPRINT\_ONLINE\_STRU\_EXT\_INF Param);

【Parameter】CchexHandle：Successfully create the handle by CChex\_Start. Input [Parameter]

DevIdx: Index of device, Input [Parameter]

Param: CCHEX\_ADD\_FINGERPRINT\_ONLINE\_STRU\_EXT\_INF

[StructLayout(LayoutKind.Sequential, Size = 6, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_ADD\_FINGERPRINT\_ONLINE\_STRU\_EXT\_INF

        {

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[]   EmployeeId;  // 5 bytes

            public byte     BackupNum; // The face template do not need setup the parameter

        }

### Response Interface

【Return value】1: The command was executed successfully; Minus value execute command failure

【Actual Data】The type of CChex\_Update return as CCHEX\_RET\_ADD\_PICTURE\_FINGERPRINT\_ONLINE\_TYPE //161

Data Part：CCHEX\_DOWNLOAD\_FACE\_PICTURE\_MODULE

[StructLayout(LayoutKind.Sequential, Size = 20, CharSet = CharSet.Ansi), Serializable]

        public struct CCHEX\_DOWNLOAD\_FACE\_PICTURE\_MODULE

        {

            public uint MachineId;

            public int Result;

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 5)]

            public byte[] EmployeeId;                      //Employee ID

            public byte Backup;                           //Backup ID

            [MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 2)]

            public byte[] padding;        // Invalid Data to fill in the structure

            public uint DataLen;          //The length of the face template data

            //[MarshalAsAttribute(UnmanagedType.ByValArray, SizeConst = 0)]

            //public byte[] PictureData;               //Face template data

        }

### Sample

int ret =CCHex\_AddFingerprintOnline\_FacePicture(CchexHandle, DevIdx, ref Param);

### Notice

Please make sure the CChex\_Start has been started successfully before running.

# 4 Return value

1：Success；

-1：Parameter Error；

-2：Lack of device resources；

-3：Unknown Error；