|  |  |
| --- | --- |
| struct | ***属性*** |
| **\_SubImageInfo** | *char StoragePath[257];*  *char type[4];*  *char ShotTime[15];*  *int Width;*  *int Height;*  *char filePath[200];*  *unsigned char \*imageData;*  *int imageSize;*  *char imageNum[6];* |
| **\_FaceInfo** | *char DeviceID[21];*  *int LeftTopX;*  *int LeftTopY;*  *int RightBtmX;*  *int RightBtmY;*  *int AccompanyNumber;*  *char LocationMarkTime[15];*  *char FaceAppearTime[15]*  *char FaceDisAppearTime[15];*  *char faceNum[6]*  *SubImageInfo imageInfo;*  *SubImageInfo faceImageInfo;*  *long id, vsid;*  *float distance, score;* |
| **\_vehicleinfo** | *int LeftTopX;*  *int LeftTopY;*  *int RightBtmX;*  *int RightBtmY;*  *char PlateNo[MAX\_PLATE\_LEN];*  *char PlateColor[MAX\_COLOR\_TYPE\_LEN];*  *char PlateReliability[3];*  *char PlateCharReliability[64];*  *SubImageInfo img\_full;*  *SubImageInfo img\_plate;* |

void set\_gat1400\_appname(const char \*app\_name);

gboolean setup\_gat1400(char \*deviceID, char \*serverip, char \*user, char \*password, char \*up\_user, char \*up\_pwd);

gboolean clean\_gat1400();

gboolean plc\_register();

gboolean plc\_unregister();

gboolean plc\_keepalive();

gboolean plc\_faceUpload(FaceInfo \*face);

gboolean plc\_vehicleUpload(VehicleInfo \*vehicle);

gboolean hk\_connect();

gboolean img\_register();

int hk\_img\_upload(const char \*bufdata, unsigned int bufsize, char \*download\_url, unsigned int url\_len);

define

|  |  |
| --- | --- |
| REGISTER | http://%s/VIID/System/Register |
| SERVERIP | 120.55.20.214 |
| UNREGISTER | http://%s/VIID/System/UnRegister |
| KEEPALIVE | http://%s/VIID/System/Keepalive |
| FACES | http://%s/VIID/Faces |
| VEHICLES | http://%s/VIID/MotorVehicles |
| IMGSVR | http://%s/VIID/ImageServer?DeviceID=%s |
| IMGSEND | http://%.64s:%.3d/VIID/Images |
| IMGREGISTER | http://%s:%d/VIID/System/Register |

结构体

|  |  |
| --- | --- |
| \_ImgSvrInfo | gchar upload\_ip[128];  int upload\_port;  gchar download\_ip[128];  int download\_port;  gchar pool[128];  gchar requrl[128];  gchar status[128];  gchar code[128]; |
| \_BestNode | gchar ip[64]; gchar token[64]; gchar requrl[64]; gchar status[64]; int port; int code; |

变量

|  |  |
| --- | --- |
| app\_name | static char app\_name[32]; |
| \_deviceID | static gchar \*\_deviceID = NULL; |
| \_register\_url | static gchar \*\_register\_url = NULL; |
| \_unregister\_url | static gchar \*\_unregister\_url = NULL; |
| \_keepalive\_url | static gchar \*\_keepalive\_url = NULL; |
| \_faces\_url | static gchar \*\_faces\_url = NULL; |
| \_vehicles\_url | static gchar \*\_vehicles\_url = NULL; |
| \_username | static gchar \_username[64]; |
| \_passward | static gchar \_password[64]; |
| \_upload\_user | static gchar \_upload\_user[64]; |
| \_upload\_pwd | static gchar \_upload\_pwd[64]; |
| \_hk\_img\_server | static gchar \*\_hk\_img\_server = NULL; |
| \_hk\_img\_send\_url | static gchar \_hk\_img\_send\_url[128]; |
| g\_img\_upload\_resp\_buf | char g\_img\_upload\_resp\_buf[GAT\_RESP\_SIZE]; |
| \_hk\_img\_svr | ImageServerInfo \_hk\_img\_svr; |
| \_hk\_best\_node | BestNodeInfo \_hk\_best\_node; |

方法

|  |
| --- |
| void set\_gat1400\_appname(const char \*application\_name) |
| static gboolean curl\_service(const char \*url, char \*buffer); |
| static gboolean curl\_service(const char \*url, char \*buffer) |
| static gboolean curl\_service\_get(const char \*url, char \* req\_body, char (\*header\_list)[128], int header\_cnt) |
| gboolean setup\_gat1400(char \*deviceID, char \*serverip, char \*user, char \*password, char \*up\_user, char \*up\_pwd) |
| gboolean clean\_gat1400() |
| gboolean plc\_register() |
| gboolean plc\_unregister() |
| gboolean plc\_keepalive() |
| gboolean plc\_faceUpload(FaceInfo \*face) |
| static void get\_time\_str(gchar\* strtime, unsigned int strlen) |
| static void get\_image\_id(char \*dest, unsigned int destlen, char \*timestr, char \*num) |
| static void getVehicleID(char id[49], const char srcID[42], const char objID[6]) |
| gboolean plc\_vehicleUpload(VehicleInfo \*vehicle) |
| static int load\_hk\_server\_info(const char \*str) |
| static gboolean hk\_get\_img\_server(const char \*url) |
| gboolean hk\_img\_register(const char \*url) |
| gboolean hk\_connect() |
| gboolean img\_register() |
| static size\_t hk\_cb\_data\_writer(void\* data, size\_t size, size\_t nmemb, void\* userdata) |
| static int curl\_img\_upload(const char \*infodata, const char \*bufdata, unsigned int bufsize, char \*img\_url, unsigned int img\_url\_len) |
| int hk\_img\_upload(const char \*bufdata, unsigned int bufsize, char \*download\_url, unsigned int url\_len) |
|  |
|  |
|  |