2014

Perio TcpReader yazılım geliştiriciler için kullanım kılavuzu

Kartlı Geçiş ve HGS Sistemi

Bu belge ART Elektronik tarafından yazılım geliştiriciler için hazırlanmıştır.

ART
ART Elektronik
01.01.2014

İçindekiler Tablosu

Bağlantı Ayarları

Cihazla bağlantı sağlamak için gerekli olan tipler aşağıdaki gibidir. Bu tipler standart değerler alabilir. Belirlenen tipler dışında değerler alamazlar.

TReaderType, TProtocolType , TDFType

Cihaza bağlanmadan önce "rdr" nesnesinin parametreleri belirtilmelidir.

RDR nesnesinin parametrik yapıları		
İsmi	**Protokol Tipi**	**Alabileceği Değerler**
ProtocolType	TProtocolType	PR0, PR1, PR2, PR3
DFType	TDFType	rdr26M, rdr63M_V3,rdr63M_V5
ReaderType	TReaderType	df4MB,df8MB
Ip	String	
Port	String	
TimeOut	Integer	
DeviceLoginKey	String	
CommandRetry	Integer	
AutoConnect	boolean	
AutoRxEnabled	boolean	

Parametre Anlamları
ProtocolType: *Cihazla bağlantı sağlamak için kullanılan protokol tipidir. Tipler yukarıda belirtilmiştir.*
DFType: *Cihazın donanım ayarlarından* ***"flash data"*** *kartını temsil eder.*
ReaderType: *Cihazın okuma tipi ayarlamak için kullanılır.*
IP: *Cihazın ile haberleşmek için kullaılan Ip adresidir. Bu Ip adresi Uniq olmalıdır. Aynı Ip adresi başka bir cihazda kullanılamaz.*
Port: *Cihaz ile haberleşmek için kullanılan port ayarıdır.*
TimeOut: *Cihaz ile bağlantı kurmak için kullanılan zaman birimdir.*
DeviceLoginKey: *Cihaz ile bağlantı sağlamak ve bilgi alışverişinde bulunmak için kullanılan cihaz şifredir.*
CommandRetry: *Cihaz*
AutoConnect: *Cihaz ile olan bağlantının yeniden bağlantı özelliğidir. Bağlantı, herhangi bir nedenden dolayı kesilirse yeniden bağlanmayı deneyecektir.*
AutoRxEnabled: *Online sistemlerde gelen komutu yakalayan ve işleyen "thread" yapının açık olup olmaması ayarıdır. Offline sistemlerde kullanılmasının gereği yoktur.*

Cihaz parametreleri atandıktan sonra "***Connect***" komutu ile bağlantı sağlanır.

(Cihazla bağlantı sağlamadan önce gerekli kontrollerin yapılması önerilir. Örneğin İP adresin gerçekten bir ip adresi olup olmadığı kontrol edilmesi programcının yararına olacaktır.)

Delphi

```
Rdr.ProtocolType := PR0; // PR0, PR1, PR2, PR3
Rdr.ReaderType := rdr63M_V3; //rdr26M, rdr63M_V3,rdr63M_V5
Rdr.DFType := df8MB; // df4MB,df8MB
Rdr.IP := edtConnectIp.Text;
Rdr.Port := edtConnectPortNo.Text;
Rdr.TimeOut := edtConnectTimeOut.Value;
Rdr.DeviceLoginKey := edtConnectKey.Text;
Rdr.CommandRetry := edtConnectCmdRtry.Value;
Rdr.AutoConnect := cbConnectAutoConnect.Checked;
Rdr.AutoRxEnabled := cbConnectAutoRxEnabled.Checked;
Rdr.Connect;
```

C#

```
rdr.pProtocolType = TProtocolType.PR3; //PR0, PR1, PR2, PR3
rdr.ReaderType = TReaderType.rdr63M_V3; // //rdr26M, rdr63M_V3,rdr63M_V5
rdr.DFType = TDFType.df8MB; //df4MB,df8MB
rdr.IP = edtIp.Text;
rdr.Port = (int)edtPortNo.Value;
rdr.TimeOut = (int)edtTimeOut.Value;
rdr.DeviceLoginKey = edtDeviceKey.Text;
rdr.CommandRetry = (int)edtCmtRetry.Value;
rdr.AutoConnect = cbAutoConnect.Checked;
rdr.AutoRxEnabled = cbAutoRxEnabled.Checked;
rdr.Connect();
```

Rdr(delphi) rdr(C#) nesnesi bağlantı sağlandıktan sonra geriye boolean tipinde cevap döndürür. Programcı isterse bağlantı olup olmadığını geri dönen değer ile kontrol edebilir.

Delphi

```
if Rdr.Connected then begin
showMessages('Cihazla bağlantı sağlandı');
end else begin
showMessages('Cihazla bağlantı sağlanamadı');
end;
```

C#

```
 if (rdr.Connected)
  {MessageBox.Show("Cihazla bağlantı sağlandı");}
  else
  {MessageBox.Show("Cihazla bağlantı sağlanamadı");}
```

Programcı dilediği zaman Rdr(delphi) rdr(C#) nesnesine “disconnect” komutu ile bağlantıyı kesebilir.

Delphi

```
Rdr.DisConnect;
```

C#

```
rdr.DisConnect();
```

Cihaz Firmware Versiyonu.

Cihaz Firmware versiyonunu öğrenmek için “***rdr***” nesninin "***GetFwVwersion***" fonksiyonu çağırılmalıdır. "***GetFwVwersion***" string türünde değer döndürmektedir.

Delphi

```
showMessages(Rdr.GetFwVwersion);
```

C#

```
MessageBox.Show(Rdr.GetFwVwersion());
```

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Cihaz Head / Teal bilgileri.

Cihaz Head/Teal bilgilerinin elde edilmesi için ***rdr*** nesnesinin "***GetHeadTailCapacity***" fonksiyonu çağırılmalıdır.

GetHeadTailCapacity Parametrik Yapısı			
Delphi		C#	
Dönen parametre	Türü	**Dönen parametre**	Türü
Head	LongWord	**Out Head**	int
Tail	LongWord	**Out Tail**	int
Capacity	LongWord	**Out Capacity**	int

Head / Tail: Giriş çıkış bilgilerinin göstergecidir. *"**Head** "* cihaza okutulan kart sayısını verir. ***Tail*** cihaz giriş / çıkış tabındaki transfer edilen kart sayısını verir. Kapasite ise cihazın kayıt kapasitesini gösterir. "***GetHeadTailCapacity***" fonksiyonu "***Head***" , "***Tail***" ve "***Capacity***" adında "***LongWord***" (***delphi***) "***int***" ***(C#)*** tipinde verileri döndüren bir metot yapısına sahiptir. Cihazdan dönen değerleri "***Head***, ***Tail***, ***Capacity***" nesnelerine atar. Fonksiyonu çağırmadan önce değerlerin atılacağı değişkenleri tanımlamalısınız. Fonksiyondan dönen değer *"**boolean"*** türündedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Head, Tail, Capacity: LongWord;
Rdr.GetHeadTailCapacity(Head, Tail, Capacity);
```

C#

```
int Head;
int Tail;
int Capacity;
rdr.GetHeadTailCapacity(out Head,out  Tail,out Capacity);
```

Örneklerde; dönen değerler değişken bloklarında tanımlanan değişkenlere atanmıştır.
Cihaza Head ve Tail bilgisini gönderen ve set eden fonksiyon "***SetHeadTail***" fonksiyonudur. Fonksiyondan dönen değer "***Boolean***" türündedir. "***SetHeadTail***" fonksiyonu "***Head***" ve "***Tail***" adında iki parametre ile çalışır.

Delphi

```
Rdr.SetHeadTail(integerDeger, integerDeger);
```

C#

```
int Head;
int Tail;
int Capacity;
rdr.GetHeadTailCapacity(out Head,out  Tail,out Capacity);
```

Tarih saat bilgileri.

Cihaza tarih saat bilgisi gönderen fonksiyon "***SetDateTime***" fonksiyonudur. Gönderilen data "***Datetime***" türünde ve "***dd.mm.yyyy***" "***hh:mm:ss***" formatında olmalıdır. Her iki bilgi birleştirilerek uygun tarih saat formatına çevrilmeli ve daha sonra gönderilmelidir. Fonksiyondan dönen değer ***boolean*** türündedir.

Delphi

```
Var
DT: TDateTime;
isSet:boolean;
DT := DateOf(edtDate.Date) + TimeOf(edtTime.time);
isset:=Rdr.SetDateTime(DT);
```

C#

```
Boolean isSet;
isSet=rdr.SetDateTime(edtDateTime.Value);
```

Programcı her iki örnekte olduğu gibi "***isSet***" değişkeninin son durumunu kontrol ederek tarih saat bilgilerinin gönderilip gönderilmediğini kontrol edebilir.

Cihaz durum bilgisi alma.

Cihaz durum bilgisi alma fonksiyonu "***GetDeviceStatus***" fonksiyonudur.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
isActive:boolean;
isActive:=Rdr.GetDeviceStatus;
```

C#

```
boolean:isActive;
isActive=rdr.GetDeviceStatus();
```

Programcı her iki örnekte olduğu gibi "***isActive***" değişkenini kontrol ederek cihazın durumu hakkında bilgi edinebilir. "***isActive***" değişkeninin değeri "***true***" ise cihazdan bilgiler alındı, "***false***" ise cihazdan bilgiler getirilemedi anlamına gelir.

Cihaz durum bilgisi, cihazın aktif olup olmadığı anlamına gelir. Cihaz aktif olmadığı durumlarda kart okuma yapılamaz. Aktif olduğu durumlarda ayarlanan “offline” ya da “online” moda göre işlemleri yürütür.

Cihaz durum bilgisi gönderme.

Cihaz durum bilgisi alma fonksiyonu ***"SetDeviceStatus"*** fonksiyonudur. Fonksiyon "***Boolean***" tipinde bir değer ile tetiklenir. "***True***" değeri cihaza aktif olması gerektiği, "***False***" değeri cihazın pasif olması gerektiği bilgisi gönderir. Bu fonksiyon "***Boolean"*** tipinde bir değer döndürür. Başarılı durumda "***true***", başarısız durumunda "***false***" döndürür.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
isActive:boolean;
isActive:=SetDeviceStatus(true);
```

C#

```
boolean:isActive;
isActive=rdr.SetDeviceStatus(true);
```

Programcı her iki örnekte olduğu gibi "***isActive***" değişkenini kontrol ederek cihazın durumu hakkında bilgi edinebilir."***isActive***" değişkeninin değeri "***true***" ise cihaza bilgi gönderildi ve aktif edildi, "***false***" ise cihaza bilgi gönderilemedi anlamına gelir.

Cihaz yeniden başlatma.

Cihaz yeniden başlatma “***rdr***” componentinin bir olayıdır. Yeniden başlatma "***rdr***" componentinin "***Reboot***" özelliği tetiklenerek meydana gelir. Bu tetikleme işlemi gerçekleştikten sonra cihazla olan bağlantının kesilmesi önerilir. Eğer program açılıp kapanmayacaksa "***reboot***" işlemi tetiklendikten sonra programı bekletmede yarar olacaktır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.Reboot;
Rdr.DisConnect;
Sleep(100);
Rdr.Connect;
```

C#

```
rdr.Reboot();
rdr.DisConnect();
Thread.Sleep(300);
rdr.Connect();
```

Cihazı fabrika ayarlarına döndürme.

Cihazı fabrika ayarlarına döndürme fonksiyonu "***SetDeviceFactoryDefault***" fonksiyonudur. Bu fonksiyon "***boolean***" tipinde bir parametre ile çalışır. Bu parametre ile (True, False) cihazın IP adresinin de fabrika ayarlarına döndürülüp, döndürülmeyeceği bilgisi iletilir. True değeri Ip adresini de fabrika ayarlarına döndürür. "Ip adresi resetleme" parametresi opsiyonel değildir. "***True***" ya da "***False***" değeri gönderilmesi zorunludur. Fonksiyondan geri dönen değer Boolean tipinde olup fabrika ayarlarına dönme başarılıysa "***true***" değilse "***false***" değeri döndürür.

Delphi

```
var
isActive:boolean;
isActive:=Rdr.SetDeviceFactoryDefault(False); // Ip adresini resetlemek istemiyorum.
isActive:=Rdr.SetDeviceFactoryDefault(True); // Ip adresini resetlemek istiyorum.
```

C#

```
boolean isActive;
isActive=Rdr.SetDeviceFactoryDefault(False); // Ip adresini resetlemek istemiyorum.
isActive=Rdr.SetDeviceFactoryDefault(True); // Ip adresini resetlemek istiyorum.
```

Programcı her iki örnekte olduğu gibi "***isActive***" değişkenini kontrol ederek cihazın durumu hakkında bilgi edinebilir."***isActive***" değişkeninin değeri "***true***" ise cihaz başarıyla fabrika ayarlarına döndürüldüğü, "***false***" ise cihaz fabrika ayarlarına döndürülemediği anlamına gelir.

Cihaz genel ayarlarının alınması.

Cihaz genel ayarlarının alınması için çağırılacak fonksiyon "***GetDeviceGeneralSettings***" fonksiyonudur. Bu fonksiyon "***TGeneralDeviceSettings***" tipinde bir class ile çağırılmalıdır. Bu sınıf dizisi cihaza ait bilgileri barındırır. Cihazdan gelen bilgiler bu dizi içerisindeki elemanlara atanır. Bu fonksiyondan dönen değer "***boolean***" tipinde bir değerdir. "***True***" olması durumunda cihaz bilgileri başarıyla alındı, "***False***" olması durumunda cihaz bilgileri alınamadı anlamına gelmektedir.

TGeneralDeviceSettings Parametrik Yapısı			
Delphi		**C#**	
DefaultScreenTxt1	String	DefaultScreenTxt1	public string
DefaultScreenTxt2	String	DefaultScreenTxt2	public string
CardReadBeepTime	byte	CardReadBeepTime	public byte
TrOut1Type	TrOutType	TrOut1Type	public TrOutType
TrOut2Type	TrOutType	TrOut2Type	public TrOutType
IdleScreenType	TIdleScreenType	IdleScreenType	public TIdleScreenType
DailyRebootEnb	Boolean	DailyRebootEnb	public Boolean
RebootTime	TTime	RebootTime	public DateTime
DevNo	word	DevNo	public ushort
Backlight	word	Backlight	public ushort
Contrast	word	Contrast	public ushort
CardReadTimeOut	word	CardReadTimeOut	public ushort
VariableClearTimeout	word	VariableClearTimeout	public ushort
RFU	array [0 .. 10] of byte		

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Bu fonksiyon "***TGeneralDeviceSettins***" tipinde bir nesne atanarak kullanılır. Fonksiyon çağırılmadan önce "***TGeneralDeviceSettins***" tipinde bir nesne oluşturulmalıdır. Fonksiyondan dönen değerler bu nesnenin elemanlarına atanarak kullanılır. Her dizi elemanın tipleri yukarıda belirtilmiştir.

TGeneralDeviceSettings Parametrik Yapısı			
Delphi		**C#**	
DefaultScreenTxt1	String	DefaultScreenTxt1	public string
DefaultScreenTxt2	String	DefaultScreenTxt2	public string
CardReadBeepTime	byte	CardReadBeepTime	public byte
TrOut1Type	TrOutType	TrOut1Type	public TrOutType
TrOut2Type	TrOutType	TrOut2Type	public TrOutType
IdleScreenType	TIdleScreenType	IdleScreenType	public TIdleScreenType
DailyRebootEnb	Boolean	DailyRebootEnb	public Boolean
RebootTime	TTime	RebootTime	public DateTime
DevNo	word	DevNo	public ushort
Backlight	word	Backlight	public ushort
Contrast	word	Contrast	public ushort
CardReadTimeOut	word	CardReadTimeOut	public ushort
VariableClearTimeout	word	VariableClearTimeout	public ushort
RFU	array [0 .. 10] of byte		

Parametre Anlamları
DefaultScreenTxt1: *Cihaz ekranının birinci satırında varsayılan olarak yazılmak istenen mesaj metinidir.* ***DefaultScreenTxt1:*** *Cihaz ekranının ikinci satırında varsayılan olarak yazılmak istenen mesaj metni ya da logo alanıdır.* ***CardReadBeepTime:*** *Cihazın kart okuduktan sonraki, buzzer çalma süresidir. 10/1 ms olarak ayarlanır. Bu değer 0 verilirse buzzer çalmayacaktır.* ***TrOut1Type:*** *Transistör çıkışının türüdür. Tetiklendiğinde verilen süre boyunca akım gönderilir veya akım kesilir. "NormalOpen" ve "NormalClosed" şeklinde iki türü vardır. "NormalOpen" durumunda kapı veya geçiş ünitesi bir mekanizma ile kapalı konumdadır. Kapıyı veya geçiş ünitesini açmak için akım gönderilir ve kapı bir geçişlik sürede açık kalır. "NormalClosed" şeklinde ise kapı veya geçiş ünitesi, sürekli olarak gönderilen bir akım ile kapalı konumda tutulur. Kapıyı veya geçiş ünitesini açmak için belli bir süre gönderilen akımı kesmek gerekir. Kapı veya geçiş ünitesinin çalışma prensipine göre, belirtilen tiplerden birini belirtmek gerekir.* ***IdleScreenType:*** *Cihazın ekranında logo ya da yazı tercihidir.* ***DailyRebootEnb:*** *Günlük yeniden başlatma süresini aktif eder.* ***"RebootTime"*** *değeri ile birlikte çalışır.* ***RebootTime:*** *Günlük yeniden başlama süresini belirtir. "****DailyRebootEnb"*** *özelliği ile birlikte çalışır. Cihaz belirtilen süre dolduğunda yeniden başlatılacaktır.* ***DevNo:*** *Cihazın numarasını temsil eder.* ***Backlight:*** *Cihazın arka plan ışığını ayarlamak için kullanılır.* ***Contrast:*** *Cihazın kontras ayarını yapmak için kullanılır.* ***CardReadTimeOut:*** *Cihazın kart okuma süresini belirtir. Cihaz belirtilen bu süre geçmeden aynı veya farklu kartları okumayacaktır.* ***VariableClearTimeout:*** *Cihazın aynı kartı tekrar okuma zaman aralığını belirler. Cihaz aynı kartı; belirtilen bu süre dolmadan okumayacaktır.*

Delphi

```
Var
rSettings: TGeneralDeviceSettins;
isSet:boolean;
isSet:=Rdr.GetDeviceGeneralSettings(rSettings);
if isSet=true then begin
... :=rSettings.DevNo;
... :=rSettings.IdleScreenType;
... :=rSettings.DefaultScreenTxt1;
... :=rSettings.DefaultScreenTxt2;
 ... :=rSettings.Backlight;
 ... :=rSettings.Contrast;
 ... :=rSettings.TrOut1Type;
 ... :=rSettings.TrOut1Type.NrOpen;
 ... :=rSettings.TrOut1Type.NrClosed;
 ... :=rSettings.TrOut2Type.NrOpen;
 ... :=rSettings.TrOut2Type.NrClosed;
 ... :=rSettings.DailyRebootEnb;
 ... :=rSettings.RebootTime;
 ... :=rSettings.CardReadBeepTime;
 ... :=rSettings.CardReadTimeOut;
 ... :=rSettings.VariableClearTimeout;
 end;
```

C#

```
TGeneralDeviceSettins rSettings;
if (rdr.GetDeviceGeneralSettings(out rSettings) == true  )
 {
... :=rSettings.DevNo;
... :=rSettings.IdleScreenType;
... :=rSettings.DefaultScreenTxt1;
... :=rSettings.DefaultScreenTxt2;
... :=rSettings.Backlight;
... :=rSettings.Contrast;
... :=rSettings.TrOut1Type;
... :=rSettings.TrOut1Type.NrOpen;
... :=rSettings.TrOut1Type.NrClosed;
... :=rSettings.TrOut2Type.NrOpen;
... :=rSettings.TrOut2Type.NrClosed;
... :=rSettings.DailyRebootEnb;
... :=rSettings.RebootTime;
... :=rSettings.CardReadBeepTime;
... :=rSettings.CardReadTimeOut;
... :=rSettings.VariableClearTimeout;
 }
```

Cihaz genel ayarlarının gönderilmesi.

Cihaz genel ayarlarının gönderilmesi için çağırılacak fonksiyon "***SetDeviceGeneralSettings***" fonksiyonudur. Bu fonksiyon "***TGeneralDeviceSettins***" tipinde bir class ile çağırılmalıdır. Bu sınıf dizisi cihaza ait bilgileri barındırır. Cihaza gönderilecek bilgiler bu dizi içerisindeki elemanlara atanmalıdır. Bu fonksiyondan dönen değer "***boolean***" tipinde bir değerdir. "***True***" olması durumunda cihaza bilgiler başarıyla gönderildi, "***False***" olması durumunda cihaza bilgiler gönderilemedi anlamına gelmektedir. Bu fonksiyon "***TGeneralDeviceSettins***" tipinde bir nesne atanarak kullanılır. Fonksiyon çağırılmadan önce "***TGeneralDeviceSettins***" tipinde bir nesne oluşturulmalıdır. Sınıf dizileri içerisindeki elemanlara değer atanarak kullanılır. Her dizi elemanın tipleri yukarıda belirtilmiştir.

Delphi

```
Var
rSettings: TGeneralDeviceSettins;
rSettings.DevNo := ....;
rSettings.IdleScreenType := stText; // yada stLogo
rSettings.DefaultScreenTxt1 := .....;
rSettings.DefaultScreenTxt2 := .....;
rSettings.Backlight := .....;
rSettings.Contrast := .....;
rSettings.TrOut1Type := NrOpen; //yada NrClosed
rSettings.TrOut2Type := NrOpen; //yada NrClosed
rSettings.DailyRebootEnb :=.....;
rSettings.RebootTime := true; // yada false;
rSettings.CardReadBeepTime := .....;
rSettings.CardReadTimeOut := .....;
rSettings.VariableClearTimeout :=....;
 if Rdr.SetDeviceGeneralSettings(rSettings) then begin
showMessage('Cihaz ayarları tamamlandı.');
end  else begin
showMessage('Cihaz ayarları tamamlanamadı.');
End;
```

C#

```
TGeneralDeviceSettins rSettings;
rSettings=new TGeneralDeviceSettins();
rSettings.DevNo = (ushort)(.....); // ushort tipinde davran
rSettings.IdleScreenType = TIdleScreenType.stText; // yada TIdleScreenType.stLogo;
rSettings.DefaultScreenTxt1 = .....;
rSettings.DefaultScreenTxt2 = edtIkinciSatir.Text;
rSettings.Backlight = (ushort).....; // ushort tipinde davran
rSettings.Contrast = (ushort)......; // ushort tipinde davran
rSettings.TrOut1Type = TrOutType.NrOpen;  // yada TrOutType.NrClosed;
rSettings.TrOut2Type = TrOutType.NrOpen; // yada TrOutType.NrClosed;
rSettings.DailyRebootEnb = true; // yada false;
rSettings.RebootTime = ....;
rSettings.CardReadBeepTime = (byte).....; // byte tipinde davran
rSettings.CardReadTimeOut = (ushort)....; // ushort tipinde davran
rSettings.VariableClearTimeout = (ushort)......; //ushort tipinde davran
 if (rdr.SetDeviceGeneralSettings(rSettings) == true)
{MessageBox.Show("Bilgiler başarıyla gönderildi.");}
else
{MessageBox.Show("Bilgiler gönderilemedi."); }
```

Cihaz çalışma modu ayarlarının alınması.

Cihaz çalışma modu ayarlarını alma fonksiyonu "***GetDeviceWorkModeSettings***" fonksiyonudur. Bu fonksiyon "***TWorkModeSettings***" tipinde bir sınıf dizisi parametresi alır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "***True***" değeri bilgilerin cihazdan başarıyla alındığını "***False***" değeri cihazdan bilgilerin getirilemediği anlamına gelir.

TWorkingMode Parametrik Yapıları			
Delphi	Değerler	**C#**	Değerler
TWorkingMode = (wmOffline,wmOfflineCard,wmTCPOnline,wmUDPOnline);		*public enum TWorkingMode {wmOffline,wmOfflineCard,wmTCPOnline,wmUDPOnline};*	
WorkingMode	TWorkingMode	WorkingMode	public TWorkingMode
OfflineModePermission	Boolean	OfflineModePermission	public Boolean
ServerAnswerTimeOut	LongWord	ServerAnswerTimeOut	public uint
RFU	array [0 .. 5] of byte		

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Cihazdan gelen bilgiler bu sınıf dizisi içindeki elemanlara atılır. Dizi elemanlarının tipleri yukarıda belirtilmiştir. Programcı isterse fonksiyondan dönen değeri kontrol edip, bilgilerin gönderilip gönderilmediğine dair bilgi sahibi olur.

Parametre Anlamları
TWorkingMode: *Cihazın çalışma modunu belirtir. "**wmOffline**", "**wmOfflineCard**", "**wmTCPOnline**", "**wmUDPOnline**" olmak üzere 4 çeşit çalışma modu vardır.* • ***wmOffline****: Cihaz sadece kendisi içerisinde tanıtılan bilgilerle çalışır. Kart okuma, giriş çıkış tetiklemeleri vb cihazda kayıtlı bilgiler ile yapar.* • ***wmOfflineCard****: Şu an için kullanılmayan bir mod.* • ***wmTCPOnline****: Cihaz herhangi pc / server desteği ile çalışır. TCP üzerinden pc / server ile iletişime geçerek pc / server tarafında tutulan bilgileri referans alarak çalışır. " **OfflineModePermission"** aktif edildiğinde ve "**ServerAnswerTimeOut"** değeri belirlendiği durumlarda, belirlenen sürede bağlantı kurulamazsa cihaz otomatik olarak **"offline"** moda geçecektir.* • ***wmUDPOnline****: Cihaz herhangi pc / server desteği ile çalışır. TCP üzerinden pc / server ile iletişime geçerek pc / server tarafında tutulan bilgileri referans alarak çalışır. **"ServerAnswerTimeOut"** değeri belirlendiği durumlarda, belirlenen sürede bağlantı kurulamazsa cihaz otomatik olarak **"offline"** moda geçecektir.* ***OfflineModePermission:*** *Cihazın TCP ya da UDP üzerinden, pc ya da servera bağlanamaması durumunda "**offline**" moda geçip geçmeyeceği ayarlayan özelliktir.* ***ServerAnswerTimeOut:*** *Cihazın TCP ya da UDP üzerinden, pc ya da servera bağlanamaması durumunda ne kadar süre sonra **"offline"** sisteme geçeceğini bildiren değerdir. Bu değer milisaniye türünden değer alır.*

Delphi

```
Var
rSettings: TWorkModeSettings;
Rdr.GetDeviceWorkModeSettings(rSettings);
.....  := rSettings.WorkingMode;
.....  := rSettings.OfflineModePermission;
.....  := rSettings.ServerAnswerTimeOut;
```

C#

```
TWorkModeSettings rSettings = new TWorkModeSettings();
rdr.GetDeviceWorkModeSettings(out rSettings);
.... = rSettings.WorkingMode;
.... = rSettings.OfflineModePermission;
.... = rSettings.ServerAnswerTimeOut;
```

Cihaz çalışma modu ayarlarının gönderilmesi.

Cihaz çalışma modu ayarlarını alma fonksiyonu "***SetDeviceWorkModeSettings***" fonksiyonudur. Bu fonksiyon "***TWorkModeSettings***" tipinde bir sınıf dizisi parametresi alır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "***True***" değeri bilgilerin cihaza başarıyla gönderildiğini "***False***" değeri cihazdan bilgilerin gönderilemediği anlamına gelir. Sınıf dizileri hakkında bilgi yukarıda verilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

"***SetDeviceWorkModeSettings***" fonksiyonu çağrılmadan önce dizi elemanları ayarlanmalıdır. Dizi elemanlarının alabileceği değerler yukarıda belirtilmiştir.

Delphi

```
Var
rSettings: TWorkModeSettings;
rSettings.WorkingMode := wmTCPOnline // yada wmOffline, wmOfflineCard, wmUDPOnline
rSettings.OfflineModePermission :=true; //yada false
rSettings.ServerAnswerTimeOut:=100;//integer değer;
```

C#

```
TWorkModeSettings rSettings = new TWorkModeSettings();
rSettings.WorkingMode = TWorkingMode.wmTCPOnline;//yada wmOffline, wmOfflineCard, wmUDPOnline
rSettings.OfflineModePermission = true; // yada false
rSettings.ServerAnswerTimeOut = (uint)....;
rdr.SetDeviceWorkModeSettings(rSettings);
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek ayarların gönderilip gönderilmediği konusunda bilgi sahibi olabilir.

Cihaz seri numarası alma.

Cihaz seri numarasını alma fonksiyonu “***rdr***” nesnesinin bir fonksiyonudur. İlgili fonksiyonun adı "***GetSerialNumber***" isimli fonksiyondur. Bu fonksiyon “***string***” tipinde bir değer döndürür.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
..... :=Rdr.GetSerialNumber;
```

Delphi

```
..... =rdr.GetSerialNumber();
```

Cihaz seri numarası gönderme.

Cihaz seri numarasını alma fonksiyonu rdr nesnesinin bir fonksiyonudur. İlgili fonksiyonun adı "***SetSerialNumber***". Bu fonksiyon "***string***" tipinde bir parametre ile çalışır ve "***boolean***" tipinde bir değer döndürür.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.SetSerialNumber('stringDeğişken');
```

C#

```
rdr.SetSerialNumber("stringDeğişken");
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek, bilgilerin cihaza gönderilip gönderilmediğini kontrol edebilir. "***True***" değeri cihaz bilgilerinin gönderildiğini, "***False***" değeri bilgilerin gönderilemediği anlamına gelir.

Seri port ve BaudRate ayarlarının alınması.

Cihazın herhangi iki ayrı cihaza (barkod okuyucu vb) bağlanması durumunda, portun ve cihazın eşleştiği BaudRate (bağlantı hızı) ayarını temsil eder. Seri 0 ilk cihazın bağlandığı BaudRate ayarı, Seri 1 ikinci cihazın bağlandığı BaudRate ayarını temsil eder. Uygulama tipi "***Okunulan Değeri Server'a gönder***" yani 0 olduğunda kart okuma olayı (OnRxCardId) tetiklenir, "***Card Okutulmuş Gibi yap***" yani 1 olduğunda "kart okutulmuş gibi" davranır. İlgili fonksiyonun adı "***GetSerialPortSettings***" isimli fonksiyondur. Fonksiyondan geri dönen değer "***boolean***" tipindedir ve "***True***" başarılı, "***False***" başarısız anlamındadır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
SerailAppType,s0,S1:byte;
……. := s0;
……. := s1;
……. := SerailAppType;
```

C#

```
byte s0, s1, SerailAppType;
…… = s0;
…… = s1;
…… = SerailAppType;
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek, bilgilerin cihaz bilgilerinin alınıp alınmadığını kontrol edebilir. "***True***" değeri cihaz bilgilerinin alındığını, "***False***" değeri bilgilerinin alınamadığı anlamına gelir.

ReadCardBlockData Metodu.

Bu method "***OnCardRead***" eventı, yani kart okuma esnasında çalışır. Bu metod 4 parametre ile çalışmaktadır. SectorNo byte tipinde, BlockNo Byte tipinde, KeyType TkeyType tipinde, valueBuff byte dizisi tipindedir. Bu metod kartın farklı sektörlerine farklı verileri ve veri tiplerini okumak için yapılmıştır. Örneğin farklı zamanlarda ve farklı geçişlerde farklı fiyat tarifleri uygulanmak isteniyor. Bu durumda sürekli aynı sektöre yazmak istenmeyebilir. İlk yazılan sektörden bilgi almak ve diğer sektörlere yazmak gerektiği durumlarda bu olay kullanılır. Bu method "***ReadDataBlock***" isimli global değişkenin "***true***" değeri almasıyla tetiklenir. Bu değişkenin varsayılan değeri "***False***" olarak atanmıştır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

WriteCardBlockData Metodu.

Bu method "***OnCardRead***" eventı içindei yani kart okuma esnasında çalışır. Bu metod 4 parametre ile çalışmaktadır. SectorNo byte tipinde, BlockNo Byte tipinde, KeyType TkeyType tipinde, valueBuff byte dizisi tipindedir. Bu metod kartın farklı sektörlerine farklı verileri ve veri tiplerini yazmak için yapılmıştır. Örneğin farklı zamanlarda ve farklı geçişlerde farklı fiyat tarifleri uygulanmak isteniyor. Bu durumda sürekli aynı sektöre yazmak istenmeyebilir. İlk yazılan sektörden bilgi almak ve diğer sektörlere yazmak gerektiği durumlarda bu olay kullanılır. Bu method "***ReadDataBlock***" isimli global değişkenin "***true***" değeri almasıyla tetiklenir. Bu değişkenin varsayılan değeri "***False***" olarak atanmıştır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Seri port ve BaudRate ayarlarının gönderilmesi.

Fonksiyondan geri dönen değer "***boolean***" tipindedir ve "***True***" başarılı, "***False***" başarısız anlamındadır. İlgili fonksiyon "SetSerialPortSettings" isimli fonksiyondur. Bu fonksiyon 3 parametre ile çalışır. Parametreler "***Byte***" tipinde olmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
rdr.SetSerialPortSettings(byteDeğer, byteDeğer,byteDeğer);
```

C#

```
rdr.SetSerialPortBaudrateSettings((byte)değer, (byte)değer, (byte)değer);
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek, bilgilerin cihaza gönderilip gönderilmediğini kontrol edebilir. "***True***" değeri cihaz bilgilerinin gönderildiğini, "***False***" değeri bilgilerin gönderilemediği anlamına gelir.

Haberleşme ayarlarının alınması.

Haberleşme ayarlarının alınması "***GetDeviceTCPSettings***" fonksiyonu ile sağlanır. Cihazdan gelen bilgiler "***TTCPSettings***" tipinde bir sınıf dizisi elemanlarına atanırlar. Sınıf dizi elemanları ve tipleri aşağıda belirtilmiştir. Fonksiyondan dönen değer "***Boolean***" tipindedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TTCPSettings Parametrik Yapıları			
Delphi	**Değerler**	**C#**	**Değerler**
TProtocolType = (PR0, PR1, PR2, PR3);		public enum TProtocolType { PR0, PR1, PR2, PR3 };	
IPAdress	String	IPAdress	Public string
DefGetway	String	DefGetway	Public string
NetMask	string	NetMask	Public string
PriDNS	string	PriDNS	Public string
SecDNS	string	SecDNS	Public string
Port	word	Port	Public ushort
RemIpAdress	String	RemIpAdress	Public string
DHCP	Boolean	DHCP	Public Boolean
ConnectOnlyRemIpAdress	Boolean	ConnectOnlyRemIpAdress	Public Boolean
ProtocolType	TProtocolType	ProtocolType	Public TProtocolType
ServerEchoTimeOut	Byte	ServerEchoTimeOut	Public Byte

Parametre Anlamları
TProtocolType: *"PR0", "PR1", "PR2", "PR3" olmak üzere 4 adet protokol tipi vardır. Cihazın firmware sürümüne bağlı olarak haberleşme ayarını temsil eder.* ***IPAdress:*** *Cihaz ile haberleşmede kullanılacak ıp adresini temsil eder.* ***DefGetway:*** *Varsayılan ağ geçidini temsil eder.* ***NetMask:*** *Ağ maskesini temsil eder.* ***PriDNS:*** *DNS ayarını temsil eder.* ***SecDNS:*** *İkincil DNS ayarını temsil eder.* ***Port:*** *Port numarasını temsil eder.* ***RemIpAdress:*** *Sadece belli bir Ip adresten bağlanmak adına, bağlantı kurabilecek Ip adresini temsil eder. "**ConnectOnlyRemIpAdress**" özelliği ile birlikte çalışır.* ***DHCP:*** *Otomatik Ip adresi alma özelliğini temsil eder.* ***ConnectOnlyRemIpAdress:*** *Sadece belli bir Ip adresten cihazın haberleşme ayarlarına bağlanmanızı yapılandıran ayardır. **"RemIpAdress"** özelliği ile birlikte çalışır.* ***ServerEchoTimeOut:*** *Belirtilen dakika cinsinden server ya da PC ile haberleşip cihaz ile bağlantı olup olmadığının kontrolünü gerçekleştirir. Cihaz belli bir paketi gönderip aynı paketin sağlıklı bir şekilde geri gelmesiyle cihaza ulaşabildiğini anlar.*

Delphi

```
Var
rSettings: TTCPSettings;
Rdr.GetDeviceTCPSettings(rSettings);
.... :=rSettings.ProtocolType;// PR0, PR1, PR2, PR3
.... :=rSettings.IPAdress;
.... :=rSettings.NetMask;
.... :=rSettings.DefGetway;
.... :=rSettings.PriDNS;
.... :=rSettings.SecDNS;
.... :=rSettings.Port;
.... :=rSettings.RemIpAdress;
.... :=rSettings.ConnectOnlyRemIpAdress;
.... :=rSettings.DHCP;
…. := rSettings. ServerEchoTimeOut;
```

C#

```
TDeviceTCPSettings rSettings;
rdr.GetDeviceTCPSettings(out rSettings);
.... =rSettings.ProtocolType;// PR0, PR1, PR2, PR3
.... =rSettings.IPAdress;
.... =rSettings.NetMask;
.... =rSettings.DefGetway;
.... =rSettings.PriDNS;
.... =rSettings.SecDNS;
.... =rSettings.Port;
.... =rSettings.RemIpAdress;
.... =rSettings.ConnectOnlyRemIpAdress; // false yada true değer döndürür.
.... =rSettings.DHCP;// false yada true değer döndürür.
…..= rSettings.ServerEchoTimeOut;
```

Haberleşme ayarlarının gönderilmesi.

Haberleşme ayarlarının gönderilmesi "***SetDeviceTCPSettings***" fonksiyonu ile sağlanır. Cihazdan gönderilecek bilgiler "***TTCPSettings***" tipinde bir sınıf dizisi elemanlarına atanırlar. Sınıf dizi elemanları ve tipleri yukarıda belirtilmiştir. Fonksiyon çağırılmadan önce "***TTCPSettings***" tipinde bir değişken oluşturulmalıdır. Cihazdan gönderilecek bilgiler, oluşturulan değişkenin elemanlarına atanır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "True" değeri bilgilerin başarıyla alındığını "***False***" değeri bilgilerin alınamadığını bildirir. Dizi elemanlarının alabileceği değerler aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TTCPSettings Parametrik Yapıları			
Delphi	**Değerler**	**C#**	**Değerler**
TProtocolType = (PR0, PR1, PR2, PR3);		public enum TProtocolType { PR0, PR1, PR2, PR3 };	
IPAdress	String	IPAdress	Public string
DefGetway	String	DefGetway	Public string
NetMask	string	NetMask	Public string
PriDNS	string	PriDNS	Public string
SecDNS	string	SecDNS	Public string
Port	word	Port	Public ushort
RemIpAdress	String	RemIpAdress	Public string
DHCP	Boolean	DHCP	Public Boolean
ConnectOnlyRemIpAdress	Boolean	ConnectOnlyRemIpAdress	Public Boolean
ProtocolType	TProtocolType	ProtocolType	Public TProtocolType
ServerEchoTimeOut	Byte	ServerEchoTimeOut	Public Byte

Delphi

```
Var
rSettings: TTCPSettings;
rSettings.ProtocolType:=.....//PR0, PR1, PR2, PR3 değerlerinden biri olmalıdır.
rSettings.IPAdress := .....;
rSettings.NetMask := ......;
rSettings.DefGetway := ......;
rSettings.PriDNS := ......;
rSettings.SecDNS := ......;
rSettings.Port := ......;
rSettings.RemIpAdress := ......;
rSettings.ConnectOnlyRemIpAdress := true; // yada false;
rSettings.DHCP := true; // yada false;
rSettings.ServerEchoTimeOut := ……;
Rdr.SetDeviceTCPSettings(rSettings)
```

C#

```
TTCPSettings rSettings;
rSettings = new TTCPSettings();
rSettings.ProtocolType = .....; // PR0, PR1, PR2, PR3 değerlerinden biri olmalıdır.
rSettings.IPAdress = .....;
rSettings.NetMask = .....;
rSettings.DefGetway = ......;
rSettings.PriDNS = ......;
rSettings.SecDNS = .....;
rSettings.Port = (ushort)......;
rSettings.RemIpAdress = ......;
rSettings.ConnectOnlyRemIpAdress = true; // yada false
rSettings.DHCP = true; // yada false;
rSettings. ServerEchoTimeOut = ……;
rdr.SetDeviceTCPSettings(rSettings);
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek bilgilerin gönderilip gönderilmediği konusunda bilgi sahibi olabilir. "***True***" değeri bilgilerin gönderildiğini "***false***" değeri gönderilemediği anlamına gelir.

UDP Ayarlarının alınması.

UDP ayalarının alınmasını sağlayan fonksiyon "***GetDeviceUDPSettings***" isimli fonksiyondur. Bu fonksiyona "***TUDPSettings***" tipinde bir sınıf dizisi parametre olarak gönderilmelidir. Cihaz bilgileri, parametre ile gönderilen "***TUDPSettings***" dizi elemanlarına atanacaktır. "***TUDPSettings***" dizi ve elemanları aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TUDPSettings Parametrik Yapıları			
Delphi	**Değerler**	**C#**	**Değerler**
UDPIsActive	Boolean	UDPIsActive	public Boolean
UDPLogIsActive	Boolean	UDPLogIsActive	public Boolean
RemUDPAdress	String	RemUDPAdress	public String
UDPPort	word	UDPPort	public ushort

Delphi

```
Var
rSettings: TUDPSettings;
Rdr.GetDeviceUDPSettings(rSettings);
....:= rSettings.RemUDPAdress;
....:= rSettings.UDPPort;
....:= rSettings.UDPIsActive;
....:= rSettings.UDPLogIsActive;
```

C#

```
TUDPSettings rSettings;
rdr.GetDeviceUDPSettings(out rSettings);
.... = rSettings.RemUDPAdress;
.... = rSettings.UDPPort;
.... = rSettings.UDPIsActive;
.... = rSettings.UDPLogIsActive;
```

Fonksiyondan dönen değer "***Boolean***" tipindedir. "True" değeri UDP bilgilerinin başarıyla alındığını, "***False***" değeri "***UDP"*** bilgilerinin alınamadığı anlamına gelir. Programcı isterse fonksiyondan gelen değeri kontrol ederek işlem yapabilir.

UDP Ayarlarının gönderilmesi.

UDP ayalarının alınmasını sağlayan fonksiyon "***SetDeviceUDPSettings***" isimli fonksiyondur. Bu fonksiyona "***TUDPSettings***" tipinde bir sınıf dizisi parametre olarak gönderilmeden önce "***TUDPSettings***" dizi elemanlarının değerleri belirlenmelidir. "***TUDPSettings***" dizi ve elemanları aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TUDPSettings Parametrik Yapıları			
Delphi	**Değerler**	**C#**	**Değerler**
UDPIsActive	Boolean	UDPIsActive	Public Boolean
UDPLogIsActive	Boolean	UDPLogIsActive	Public Boolean
RemUDPAdress	String	RemUDPAdress	Public String
UDPPort	word	UDPPort	Public Word

Delphi

```
Var
  rSettings: TUDPSettings;
  rSettings.RemUDPAdress := .....;
  rSettings.UDPPort := ......;
  rSettings.UDPIsActive := true; // yada false;
  rSettings.UDPLogIsActive := true; // yada false;
  Rdr.SetDeviceUDPSettings(rSettings);
```

C#

```
TUDPSettings rSettings;
rSettings = new TUDPSettings();
rSettings.RemUDPAdress = .....;
rSettings.UDPPort = (ushort)....;
rSettings.UDPIsActive = true; //yada false
rSettings.UDPLogIsActive = true; //yada false;
rdr.SetDeviceUDPSettings(rSettings);
```

Fonksiyondan dönen değer "***Boolean***" tipindedir. "***True***" değeri ***UDP*** bilgilerinin başarıyla gönderildiğini, "***False***" değeri ***UDP*** bilgilerinin gönderilemediği anlamına gelir. Programcı isterse fonksiyondan gelen değeri kontrol ederek işlem yapabilir.

TCP Client Ayarlarının Alınması.

TCP Client ayarlarının alınmasını sağlayan fonksiyon "***GetDeviceClientTCPSettings***" isimli fonksiyondur. Bu fonksiyona parametre olarak "***TClientTCPSettings***" tipinde dizi gönderilmelidir. Cihazdan gelen TCP Client ayarları bu dizinin elemanlarına atanır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "***TClientTCPSettings***" elemanları ve alabileceği değerler aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TClientTCPSettings Parametrik Yapıları			
Delphi	**Değerler**	**C#**	**Değerler**
IPAdress	String	IPAdress	Public String
Port	word	Port	Public word

Delphi

```
Var
rSettings: TClientTCPSettings;
Rdr.GetDeviceClientTCPSettings(rSettings);
..... := rSettings.IPAdress;
..... := rSettings.Port;
```

C#

```
TClientTCPSettings rSettings;
rSettings = new TClientTCPSettings();
rdr.GetDeviceClientTCPSettings(out rSettings);
..... = rSettings.IPAdress;
..... = (ushort)rSettings.Port;
```

Fonksiyondan dönen değer "***Boolean***" tipindedir. "***True***" değeri TCP Client başarıyla alındığını, "***False***" değeri TCP Client alınamadığı anlamına gelir. Programcı isterse fonksiyondan gelen değeri kontrol ederek işlem yapabilir.

TCP Client Ayarlarının Gönderilmesi.

TCP Client ayarlarının gönderilmesini sağlayan fonksiyon "***SetDeviceClientTCPSettings***" isimli fonksiyondur. Bu fonksiyona parametre olarak "***TClientTCPSettings***" tipinde dizi gönderilmelidir. Cihazdan gelen TCP Client ayarları bu dizinin elemanlarına atanır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "***TClientTCPSettings***" elemanları ve alabileceği değerler aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TClientTCPSettings Parametrik Yapıları			
Delphi	**Değerler**	**C#**	**Değerler**
IPAdress	String	IPAdress	Public String
Port	word	Port	Public word

Delphi

```
Var
rSettings: TClientTCPSettings;
rSettings.IPAdress := ....;
rSettings.Port := .....;
Rdr.SetDeviceClientTCPSettings(rSettings);
```

C#

```
TClientTCPSettings rSettings;
rSettings = new TClientTCPSettings();
rSettings.IPAdress = ....;
rSettings.Port = (ushort).....;
rdr.SetDeviceClientTCPSettings(rSettings);
```

Fonksiyondan dönen değer "***Boolean***" tipindedir. "***True***" değeri TCP Client başarıyla gönderildi "***False***" değeri TCP Client gönderilmediği anlamına gelir. Programcı isterse fonksiyondan gelen değeri kontrol ederek işlem yapabilir.

Haberleşme Şifresi Gönderme İşlemi.

Haberleşme şifresi gönder işlemini gerçekleştiren fonksiyon "***SetDeviceLoginKey***" fonksiyonudur. Bu fonksiyona "***String***" tipinde bir değer gönderilmedir. Fonksiyondan dönen değer "***Boolean***" tipinde bir değerdir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.SetDeviceLoginKey('Stringİfade');
```

C#

```
Rdr.SetDeviceLoginKey('Stringİfade');
```

Fonksiyondan dönen değer "***Boolean***" tipindedir. "***True***" değeri haberleşme şifresinin başarıyla gönderildiğini, "***False***" değeri haberleşme şifresinin gönderilemediği anlamına gelir. Programcı isterse fonksiyondan gelen değeri kontrol ederek işlem yapabilir.

Mac Adresi Alma İşlemi.

Mac adresi alma işlemini gerçekleştiren fonksiyon "***GetMACAddress***" fonksiyondur. Bu fonksiyon parametre almadan "***String***" türünde bir değer döndürür.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
MACAddress:string;
MACAddress := Rdr.GetMACAddress;
```

C#

```
String:MACAddress;
MACAddress := Rdr.GetMACAddress();
```

Fonksiyondan dönen değer "*" (yıldız) karakteri ise mac adresi alınamamış anlamına gelmektedir. Programcı isterse dönen değerin "*" olup olmadığını kontrol ederek, mac adresi alma işleminin gerçekleşip gerçekleşmediğini kontrol edebilir.

Web Arayüzü Şifresi Alma.

Web arayüzü şifresi alma işlemini gerçekleştiren fonksiyon "***GetWebPassword***" fonksiyonudur. Bu fonksiyon parametresiz olarak çalışır ve "***string***" bir değer döndürür.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
pwd: String;
pwd := Rdr.GetWebPassword;
```

C#

```
string pwd="";
pwd=rdr.GetWebPassword();
if (Sifre != "*")
{MessageBox.Show(pwd);}
else
{("Web arayüzü şifresi alınamadı.");}
```

Web Arayüzü Şifresi Gönderme.

Web arayüzü şifresi gönderme işlemini gerçekleştiren fonksiyon "***SetWebPassword***" fonksiyonudur. Bu fonksiyon parametre olarak "***String***" bir değer alır. "***Boolean***" tipinde bir değer döndürür.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.SetWebPassword('StringDeğer');
```

C#

```
rdr.SetWebPassword("StringDeğer");
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek arayüz şifresinin değiştirilip değiştirilmediğini kontrol edebilir. "***True***" değeri web arayüz şifresinin değiştirildiğini, "***False***" değeri web arayüz şifresinin değiştirilemediğinin karşılığıdır.

Cihaza Mesaj Gönder (Online).

Fonksiyonun genel amacı; cihaza ait rölelerin tetiklenmesi, buzzer'ın çalma süresi ve cihazın mesaj ayarlarının yapılmasını sağlamaktır. Cihaz online mesaj yönetimini yapan fonksiyon "***SetBeepRelayAndSecreenMessage***" fonksiyonudur. Bu fonksiyon "***boolean***" değeri döndüren bir fonksiyondur. "***SetBeepRelayAndSecreenMessage***" fonksiyonu "***TLcdScreen***" tipinde geniş bir sınıf dizisinden oluşmaktadır. Bu sınıfa ait diziler, alt diziler ve dizi elemanları aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TLCDScreen Parametrik Yapısı		
	Delphi	C#
ID	word (max 2)	ushort
HeaderType	byte	byte
Caption	String (max 20)	string
Line	array [0..4] of TScreenLine	public TScreenLine[] Line *(public TLcdScreen(){this.Line = new TScreenLine[5];})*
FooterType	Byte	byte
Footer	String	string
RL_Time1	word	ushort
RL_Time2	word	ushort
BZR_time	word	ushort
IsBlink	Boolean	boolean
ScreenDuration	word	ushort
FontType	byte	byte
LineCount	byte	byte
NextScreen	word	ushort
KeyPadType	byte	byte

Parametre Anlamları
HeaderType: *0, 1, 2 değerleri alır. 0 Yok, 1 Üst Bilgi, 2 Başlık anlamlarına gelir.* ***Caption:*** *Cihazda görüntülenmesi istenen başlığı temsil eder. "**HeaderType**" olarak 2 seçilirse başlık alanı belirtilmelidir.* ***Line:*** *Cihaz ekranı maksiumum 5 satırdan oluşmaktadır. Her bir "**line**" değeri üstten başlayarak satırlar temsil etmektedir. Bu dizide tanımlanan her satırın X ve Z koordinatları vardır. Buradaki tanımlar, bu satırların başlangıç ve bitiş koordinatlarını belirler.* ***FooterType:*** *0, 1, 2 değerleri alır. 0 Yok, 1 Üst Bilgi, 2 Başlık anlamlarına gelir.* ***Footer:*** *Cihazda görüntülenmesi istenen alt başlığı temsil eder.* ***RL_Time1:*** *Okuma sonrası birinci rölenin tetikleme süresini temsil eder. Bu ayar milisaniye cinsindendir.* ***RL_Time2:*** *Okuma sonrası ikinci rölenin tetikleme süresini temsil eder. Bu ayar milisaniye cinsindendir.* ***IsBlink:*** *Kart okuma esnasında buzzerın çalma durumunu belirtir. True olduğunda buzzer aktif hale gelir. False olduğunda buzzer pasif hale gelir.* ***ScreenDuration:*** *Kart okuma esnasında cihaza gönderilen mesajın ekranda ne kadar süreyle kalacağının ayarını temsil eder.* ***FontType:*** *Okuma sonrası birinci rölenin tetikleme süresini temsil eder. Bu ayar milisaniye 10*100 milisaniye cinsindendir.* ***LineCount:*** *Ekran satır sayısını temsil eder. Ekran kullanmak istenilen* ***NextScreen:*** *Bu ayar henüz kullanılmamaktadır.* ***KeyPadType:*** *Cihazdan bilgi girişini sağlayacak klavye metodunu temsil eder. "0" küçük harf, "1" büyük harf, "2" sadece sayı girilmesini sağlar.*

Delphi

```
TScreenLine = record
Text : String;
Alligment : byte;
X : byte;
Y : byte;
end;
TLcdScreen = record
ID : word;
HeaderType : byte;
Caption : String;
Line : array [0..4] of TScreenLine; //105   +15
FooterType :Byte;
Footer : String;
RL_Time1 : word;
RL_Time2 : word;
BZR_time : word;
IsBlink : Boolean;
ScreenDuration : word;
FontType : byte;
LineCount :byte
NextScreen : word;
end;
```

C#

```
public struct TScreenLine
{ public string   Text;
 public byte Alligment
public byte X;
public byte Y;}

public class TLcdScreen
{public ushort ID;
public byte HeaderType ;
public String Caption;
public TScreenLine[] Line ;
public byte FooterType;
public String Footer;
public ushort RL_Time1;
public ushort RL_Time2;
public ushort BZR_time;
public Boolean IsBlink;
public ushort ScreenDuration;
public byte FontType;
public byte LineCount;
public ushort NextScreen;
public TLcdScreen()
{this.Line = new TScreenLine[5]; }
}
```

Fonksiyon kullanılmadan önce "**TLcdScreen**" tipinde bir değişken oluşturup, oluşturulan değişkenin elemanlarının değerleri atanmalıdır.

Delphi

SetBeepRelayAndSecreenMessage
(HeaderType,FooterType:Integer;
Caption,Text1,Text2,Text3,Text4,Text5,Footer:String;
X1,Y1,Alligment1,X2,Y2,Alligment2,X3,Y3,Alligment3,X4,Y4,Alligment4,X5,Y5,Alligment5:Byte;
LineCount,FontType:Byte;
ScreenDuration,RL_Time1,RL_Time2,BZR_time : word;
IsBlink : Boolean);

Rdr.**SetBeepRelayAndSecreenMessage**(*integerDeğer,integerDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer,byteDeğer, byteDeğer, 0, , byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer,0, byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer, wordDeğer,wordDeğer,wordDeğer, wordDeğer,booleanDeğer*);

C#

SetBeepRelayAndSecreenMessage(
int HeaderType, int FooterType, String Caption, String Text1, String Text2, String Text3, String Text4, String Text5, String Footer, Byte X1, Byte Y1, Byte Alligment1, Byte X2, Byte Y2, Byte Alligment2, Byte X3, Byte Y3,Byte Alligment3,Byte X4,Byte Y4,Byte Alligment4,Byte X5,Byte Y5,Byte Alligment5, Byte LineCount, Byte FontType, ushort ScreenDuration, ushort RL_Time1, ushort RL_Time2, ushort BZR_time, Boolean IsBlink);

rdr.SetBeepRelayAndSecreenMessage(
integerDeğer,integerDeğer,stringDeğer,stringDeğer,stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer,0, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, booleanDeğer
);

Programcı isterse fonksiyondan dönen değeri kontrol ederek cihaz ayarlarının yapılıp yapılmadığını kontrol edebilir. "***True***" değeri ayarların yapıldığının, "***False***" değeri ayarların yapılamadığının karşılığıdır.

Cihaza Mesaj Gönder (Offline).

Tanımlanan durumların herhangi bir PC bağlantısı olmadan çalışma halidir. Cihaz bu ayarlar tamamlandıktan sonra kendi içeriğinde oluşturulmuş ayarlara göre aksiyon alacaktır. Cihaz offline mesaj yönetimini yapan fonksiyon "***SetLCDMessages***" fonksiyonudur. Bu fonksiyon "***boolean***" değeri döndüren bir fonksiyondur. "***SetLCDMessages***" fonksiyonu "***TMsg***","***TScreenLine***" ve "***TLcdScreen***" sınıf dizilerinden oluşmaktadır. Detayları aşağıda belirtilmiştir. Cihazın offline ayarlarını gönderen fonksiyonu çağırmadan önce "***TLcdScreen***" dizi elemanlarına değerlerin atanması gerekmektedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TLCDScreen Parametrik Yapısı		
	Delphi	C#
ID	word (max 2)	ushort
HeaderType	byte	byte
Caption	String (max 20)	string
Line	array [0..4] of TScreenLine	public TScreenLine[] Line /*(public TLcdScreen(){this.Line = new TScreenLine[5];})*/
FooterType	Byte	byte
Footer	String	string
RL_Time1	word	ushort
RL_Time2	word	ushort
BZR_time	word	ushort
IsBlink	Boolean	boolean
ScreenDuration	word	ushort
FontType	byte	byte
LineCount	byte	byte
NextScreen	word	ushort
KeyPadType	byte	byte

Delphi

```
Var
LcdScreenMsg: TLcdScreen;
ekranKodu: Integer;
LcdScreenMsg.ID := OfffflineMsg[ekranKodu].MsgID;
LcdScreenMsg.HeaderType := byteDeğer;
LcdScreenMsg.Caption := stringDeğer;
LcdScreenMsg.Line[0].Text := stringDeğer;
LcdScreenMsg.Line[1].Text := stringDeğer;
LcdScreenMsg.Line[2].Text := stringDeğer;
LcdScreenMsg.Line[3].Text := stringDeğer;
LcdScreenMsg.Line[4].Text := stringDeğer;
LcdScreenMsg.Line[0].X := byteDeğer;
LcdScreenMsg.Line[1].X := byteDeğer;
LcdScreenMsg.Line[2].X := byteDeğer;
LcdScreenMsg.Line[3].X := byteDeğer;
LcdScreenMsg.Line[4].X := byteDeğer;
LcdScreenMsg.Line[0].Y := byteDeğer;
LcdScreenMsg.Line[1].Y := byteDeğer;
LcdScreenMsg.Line[2].Y := byteDeğer;
LcdScreenMsg.Line[3].Y := byteDeğer;
LcdScreenMsg.Line[4].Y := byteDeğer;
LcdScreenMsg.FooterType := byteDeğer;
LcdScreenMsg.Footer := stringDeğer;
LcdScreenMsg.LineCount := byteDeğer;
LcdScreenMsg.FontType := byteDeğer;
LcdScreenMsg.ScreenDuration := wordDeğer;
LcdScreenMsg.RL_Time1 := wordDeğer;
LcdScreenMsg.RL_Time2 := wordDeğer;
LcdScreenMsg.BZR_time := wordDeğer;
LcdScreenMsg.IsBlink := booleanDeğer;
Rdr.SetLCDMessages(LcdScreenMsg);
```

C#

```
TLcdScreen LcdScreenMsg;
LcdScreenMsg = new TLcdScreen();
int ekranKodu;
LcdScreenMsg.ID = (ushort)OfffflineMsg[ekranKodu].MsgID;
LcdScreenMsg.HeaderType = (byte)byteDeğer;
LcdScreenMsg.Caption = (string)stringDeğer;
LcdScreenMsg.Line[0].Text = stringDeğer;
LcdScreenMsg.Line[1].Text = stringDeğer;
LcdScreenMsg.Line[2].Text = stringDeğer;
LcdScreenMsg.Line[3].Text = stringDeğer;
LcdScreenMsg.Line[4].Text = stringDeğer;
LcdScreenMsg.Line[0].X = (byte)byteDeğer;
LcdScreenMsg.Line[1].X = (byte)byteDeğer;
LcdScreenMsg.Line[2].X = (byte)byteDeğer;
LcdScreenMsg.Line[3].X = (byte)byteDeğer;
LcdScreenMsg.Line[4].X = (byte)byteDeğer;
LcdScreenMsg.Line[0].Y = (byte)byteDeğer;
LcdScreenMsg.Line[1].Y = (byte)byteDeğer;
LcdScreenMsg.Line[2].Y = (byte)byteDeğer;
LcdScreenMsg.Line[3].Y = (byte)byteDeğer;
LcdScreenMsg.Line[4].Y = (byte)byteDeğer;
LcdScreenMsg.FooterType = (byte)byteDeğer;
LcdScreenMsg.Footer = stringDeğer;
LcdScreenMsg.LineCount =(byte)byteDeğer;
LcdScreenMsg.FontType = (byte)byteDeğer;
LcdScreenMsg.ScreenDuration = (ushort)ushortDeğer;
LcdScreenMsg.RL_Time1 = (ushort)ushortDeğer;
LcdScreenMsg.RL_Time2 = (ushort)ushortDeğer;
LcdScreenMsg.BZR_time = (ushort)ushortDeğer;
LcdScreenMsg.IsBlink = booleandeğer;
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek cihaz ayarlarının yapılıp yapılmadığını kontrol edebilir. "***True***" değeri ayarların yapıldığının, "***False***" değeri ayarların yapılamadığının karşılığıdır.

Cihaza Mesaj Gönderme ve Klavyeden Bilgi Okuma.

Herhangi bir PC ya da sunucu tarafından cihaza mesaj göndermek ve o esnada cihaza klavye yardımıyla bilgi girişini sağlayan metoddur. Bu metod aynı zamanda cihazın klavyesinden girilen bilgiyi alarak işlenmesini sağlar. Bu metod okuyucu tipi sadece "63M" versiyonlarda çalışır. İlgili fonksiyon "***SetBeepRelayAndInboxMessage***" ismiyle anılır. PC aracılığı ile cihaza komut gönderildikten ve klavyeden(cihaz) tuşlara basıldıktan sonra, "***OnInputText***" olayı tetiklenir. Bu event klavyeden girilen ifadeyi kullanmak için kullanılır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var LcdScreenMsg : TLcdScreen;
LcdScreenMsg.ID := 71;
LcdScreenMsg.HeaderType := HeaderType;
LcdScreenMsg.Caption := Caption;
LcdScreenMsg.Line[0].X := X1;
LcdScreenMsg.Line[0].Y := Y1;
LcdScreenMsg.Line[0].Alligment := Alligment1;
LcdScreenMsg.Line[0].Text := Text1;
LcdScreenMsg.Line[1].X := X2;
LcdScreenMsg.Line[1].Y := Y2;
LcdScreenMsg.Line[1].Alligment := Alligment2;
LcdScreenMsg.Line[1].Text := Text2;
LcdScreenMsg.Line[2].X := 0;
LcdScreenMsg.Line[2].Y := 0;
LcdScreenMsg.Line[2].Alligment := 0;
LcdScreenMsg.Line[2].Text := '';
LcdScreenMsg.Line[3].X := 0;
LcdScreenMsg.Line[3].Y := 0;
LcdScreenMsg.Line[3].Alligment := 0;
LcdScreenMsg.Line[3].Text := '';
LcdScreenMsg.Line[4].X := 0;
LcdScreenMsg.Line[4].Y := 0;
LcdScreenMsg.Line[4].Alligment := 0;
LcdScreenMsg.Line[4].Text := '';
LcdScreenMsg.FooterType := 1;
LcdScreenMsg.Footer := '';
LcdScreenMsg.RL_Time1 := RL_Time1;
LcdScreenMsg.RL_Time2 := RL_Time2;
LcdScreenMsg.BZR_time := BZR_time;
LcdScreenMsg.IsBlink := IsBlink;
LcdScreenMsg.ScreenDuration := ScreenDuration;
LcdScreenMsg.FontType := 0;
LcdScreenMsg.LineCount := 3;
LcdScreenMsg.NextScreen := 3;
LcdScreenMsg.KeyPadType := KeyPadType;
Result := (tcpSetBeepRelayAndInboxMessage(LcdScreenMsg)=0);
```

C#

```
TLcdScreen LcdScreenMsg = new TLcdScreen();
        try
         {
LcdScreenMsg.ID = 71;
LcdScreenMsg.HeaderType  = (byte)HeaderType;
LcdScreenMsg.Caption  = Caption;
LcdScreenMsg.Line[0].X  = X1;
LcdScreenMsg.Line[0].Y  = Y1;
LcdScreenMsg.Line[0].Alligment  = Alligment1;
LcdScreenMsg.Line[0].Text  = Text1;
LcdScreenMsg.Line[1].X  = X2;
LcdScreenMsg.Line[1].Y  = Y2;
LcdScreenMsg.Line[1].Alligment  = Alligment2;
LcdScreenMsg.Line[1].Text  = Text2;
LcdScreenMsg.Line[2].X  = 0;
LcdScreenMsg.Line[2].Y  = 0;
LcdScreenMsg.Line[2].Alligment  = 0;
LcdScreenMsg.Line[2].Text  = "";
LcdScreenMsg.Line[3].X  = 0;
LcdScreenMsg.Line[3].Y  = 0;
LcdScreenMsg.Line[3].Alligment  = 0;
LcdScreenMsg.Line[3].Text  = "";
LcdScreenMsg.Line[4].X  = 0;
LcdScreenMsg.Line[4].Y  = 0;
LcdScreenMsg.Line[4].Alligment  = 0;
LcdScreenMsg.Line[4].Text  = "";
LcdScreenMsg.FooterType  = 1;
LcdScreenMsg.Footer  =
LcdScreenMsg.RL_Time1  = RL_Time1;
LcdScreenMsg.RL_Time2  = RL_Time2;
LcdScreenMsg.BZR_time  = BZR_time;
LcdScreenMsg.IsBlink  = IsBlink;
LcdScreenMsg.ScreenDuration  = ScreenDuration;
LcdScreenMsg.FontType  = 0;
LcdScreenMsg.LineCount  = 3;
LcdScreenMsg.NextScreen  = 3;
LcdScreenMsg.KeyPadType = KeyPadType;
}
catch (Exception)
{throw;}
return  (tcpSetBeepRelayAndInboxMessage(LcdScreenMsg)==0);
```

Delphi

Rdr. **SetBeepRelayAndInboxMessage** (*integerDeğer,integerDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer,byteDeğer, byteDeğer, 0, , byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer,0, byteDeğer, byteDeğer, 0, byteDeğer, byteDeğer, wordDeğer,wordDeğer,wordDeğer, wordDeğer,booleanDeğer,* (byte)byteDeğer);

C#

rdr.SetBeepRelayAndSecreenMessage(
integerDeğer,integerDeğer,stringDeğer,stringDeğer,stringDeğer, stringDeğer, stringDeğer, stringDeğer, stringDeğer, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer,0, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer, 0, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, (byte)byteDeğer, booleanDeğer,(byte)byteDeğer);

Offline Mesajlar Fabrika Ayarına Dönme İşlemi.

Offline mesajlar fabrika ayarlarına dönme işlemi fonksiyonu "***SetLCDMessagesFactoryDefault***" isimli fonksiyondur. Bu fonksiyon herhangi bir parametreye ihtiyaç duymadan çalışır. Fonksiyondan geriye "***boolean***" tipinde bir değer döner.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
rdr.SetLCDMessagesFactoryDefault;
```

C#

```
rdr.SetLCDMessagesFactoryDefault;
```

Programcı isterse fonksiyondan dönen değeri kontrol ederek cihaz ayarlarının fabrika ayarlarına dönüp dönmediğini kontrol edebilir. "***True***" değeri fabrika ayarlarına döndüğünü, "***False***" değeri fabrika ayarlarına dönülemediğini belirtir.

Hafta Gün İsimleri Alma.

Bu adım farklı dillerde yapılan çalışmalar için tasarlanmıştır. Cihazda görüntülenecek gün isimlerinin bu fonksiyon aracılığı iler değiştirebilir ve cihazın kullandığı gün adları hakkında bilgi edinebilirsiniz. Hafta Gün İsimleri Alma fonksiyonu "***GetWeekDayNames***" fonksiyonudur. Fonksiyon "***TWeekDays***" tipinde bir sınıf dizisi yapısındaki parametre ile çalışır. "***TWeekDays***" sınıf dizisi aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
TWeekDays = record
Names : array [0..6] of string;
end;
```

C#

```
public class TWeekDays
{
public string[] names;
public TWeekDays()
{
names = new string[7];
names[0] = "Pazartesi";
names[1] = "Salı";
names[2] = "Çarşamba";
names[3] = "Perşembe";
names[4] = "Cuma";
names[5] = "Cumartesi";
names[6] = "Pazar";
}
}
```

Sınıf dizisinden dönen değerler "***TWeekDays***" yapısındaki elemanlara atanır. Dizi elemanları "***string"*** türündendir. Fonksiyondan dönen değer "***Boolean***" türündendir. "***True***" değeri bilgilerin alındığı "***False***" bilgilerin alınamadığı anlamına gelir.

Fonksiyon "***Names***" isimindeki 7 adet dizi elemanlarına sırayla değerleri atar. Programcı, 7 kerelik dönen bir döngü içersinden "***TWeekDays***" tipindeki değişkenin "***Names***" elemanlarına sırasıyla değerleri atamalıdır.

Delphi

```
var
WeekDays: TWeekDays;
Rdr.GetWeekDayNames(WeekDays);

for I := 0 to 6 do
..... := WeekDays.Names[i];
```

C#

```
var
TWeekDays WeekDays;
WeekDays = new TWeekDays();
rdr.GetWeekDayNames(out WeekDays);
for (int i = 0; i <= 6; i++)
{
.... =     WeekDays.names[i];
}
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Hafta Gün İsimleri Gönderme.

Hafta Gün İsimleri Gönderme fonksiyonu "***SetWeekDayNames***" fonksiyonudur. Fonksiyon "***TWeekDays***" tipinde bir sınıf dizisi yapısındaki parametre ile çalışır. "***TWeekDays***" sınıf dizisi yukarıda belirtilmiştir. Programcı, bir döngü ile "***TWeekDays***" tipindeki değişken oluşturmalı ve "***Names***" ismindeki 7 adet dizi elemanlarına değerleri sırayla atamalıdır. Atanacak değerler string türünde olmalıdır. Fonksiyondan dönen değer "***boolean***" türündedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
WeekDays: TWeekDays;
i: Integer;
for I := 0 to 6 do
WeekDays.Names[i] := .....;
```

C#

```
TWeekDays WeekDays;
WeekDays = new TWeekDays();
for (int i = 0; i <= 6; i++)
{
WeekDays.names[0] = .......;
}
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Uygulama Ayarları / Genel Ayarlar / Ayarları Alma.

Uygulama ayarlarında "genel ayarları" getiren fonksiyon "***GetAppGeneralSettings***" fonksiyonudur. Bu fonksiyon "***TAccessGeneralSettings***" tipinde bir sınıf dizisi ve "***TAccessMode*** " , "***TInputSettings***" tipinde alt dizilerden oluşur. Diziye ait bilgileri ve elemanları hakkında bilgiyi aşağıda bulabilirsiniz. Fonksiyon; "***TAccessGeneralSettings***" tipinde bir dizi parametresi ile çalışır. Cihazdan gelen bilgiler bu diziye ve bağlı olan alt dizilerdeki elemanlara değerleri atar.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TInputSettings Parametrik Yapısı		
	Delphi	C#
InputType	Byte	public byte
InputDurationTimeout	word	public ushort

TAccessMode Parametrik Yapısı		
	Delphi	C#
AccessType	Byte	public byte
PasswordType	Byte	public byte

TAccessGeneralSettings Parametrik Yapısı		
	Delphi	C#
InputSettings	TInputSettings	public TInputSettings
AccessMode	TAccessMode	public TAccessMode
TimeAccessConstraintEnabled	Boolean	public Boolean

Parametre Anlamları
InputType: 0, 1, 2 (“***Sadece Kart” , “Kart veya Şifre”, “Kart ve Şifre”)*** *olmak üzere 3 farklı modda çalışır.* • ***Sadece Kart:*** *Kapı veya geçiş ünitesi, tanımlı kartların cihaza gösterilmesiyle çalışır.* • ***Kart veya Şifre:*** *Kapı veya geçiş ünitesi, tanımlı kartların cihaza gösterilmesiyle ya da kullanıcıların şifre girmesiyle çalışır.* • ***Kart ve Şifre:*** *Kapı veya geçiş ünitesi, tanımlı kartların cihaza gösterilmesinin ardından şifre girilmesiyle çalışır.* ***PasswordType:*** 0,1 (“***Sadece Şifre” , “Kişi No + Şifre”)*** *olmak üzere iki farklı parametre alır.* ***“InputType”*** *olarak 1,2 (****“Kart veya Şifre”, “Kart ve Şifre”)*** *seçilirse bu parametrenin yapılandırılaması gerekir.* • ***Sadece Şifre:*** *Kullanıcı sadece kendisi için belirlenmiş şifresini girmek zorundadır.* • ***Kişi No + Şifre:*** *Kullanıcı kendisi için belirlenmiş kişi numarası ve şifresini girmek zorundadır.*

Delphi

```
var
Settings: TAccessGeneralSettings;
Rdr.GetAppGeneralSettings(Settings);
..... := Settings.AccessMode.AccessType;
..... := Settings.AccessMode.PasswordType;
..... := Settings.InputSettings.InputType;
..... := Settings.InputSettings.InputDurationTimeout;
..... := Settings.TimeAccessConstraintEnabled;
```

C#

```
TAccessGeneralSettings Settings;
Settings = new TAccessGeneralSettings();
rdr.GetAppGeneralSettings(out Settings);
..... = Settings.AccessMode.AccessType;
..... = Settings.AccessMode.PasswordType;
..... = Settings.InputSettings.InputType;
..... = Settings.InputSettings.InputDurationTimeout;
..... = Settings.TimeAccessConstraintEnabled;//
```

Yukarıdaki örnekte cihazdan gelen bilgiler "***Settings***" dizi değişkenine atanmıştır. Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Uygulama Ayarları / Genel Ayarlar / Ayarları Gönderme.

Uygulama ayarlarında "genel ayarları" gönderen fonksiyon "***SetAppGeneralSettings***" fonksiyonudur. Bu fonksiyon "***TAccessGeneralSettings***" tipinde bir sınıf dizisi ve "***TAccessMode*** " , "***TInputSettings***" tipinde alt dizilerden oluşur. Diziye ait bilgileri ve elemanları hakkında bilgiyi yukarıda bulabilirsiniz. Fonksiyon çağırılmadan önce, cihaza gönderilecek bilgilerin atanabilmesi için "***TAccessGeneralSettings***" tipinde bir dizi değişken oluşturulmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
Settings: TAccessGeneralSettings;
Settings.AccessMode.AccessType := .....;
Settings.AccessMode.PasswordType := .....;
Settings.InputSettings.InputType := ......;
Settings.InputSettings.InputDurationTimeout :=..... ;
Settings.TimeAccessConstraintEnabled := True //yada false
Rdr.SetAppGeneralSettings(Settings);
```

C#

```
TAccessGeneralSettings Settings;
Settings = new TAccessGeneralSettings();
Settings.AccessMode.AccessType = (byte).....;
Settings.AccessMode.PasswordType =(byte)....;
Settings.InputSettings.InputDurationTimeout=(ushort).....;
Settings.InputSettings.InputType=(byte).....;
Settings.TimeAccessConstraintEnabled=true; //yada false
rdr.SetAppGeneralSettings(rSettings);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Zaman Kısıt Tablosu Alma İşlemi.

Zaman kısıt tablosu, bir listeye en fazla 8 adet başlangıç ve bitiş zaman aralığı atanabilmektedir. Bu işlem sistemin belirlenen zaman aralıklarında çalışması için tasarlanmıştır. Zaman kısıt tablosu; ***65*** adet farklı liste verisi tutabilmektedir. Her listeye en fazla 8 adet başlangıç ve bitiş zamanı atanabilmektedir. Zaman kısıt tablosu almak işlemini gerçekleştiren fonksiyon "***GetTimeConstraintTables***" fonksiyonudur. Bu fonksiyon iki parametre ile çalışır. Birinci parametre "***byte***" tipinde olup "***Tablo numarasını***" temsil etmekte, ikinci parametre "***TTACList***" tipinde bir sınıf dizisidir. "***TTACList***" sınıf dizisi aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TTACList Parametrik Yapısı		
	Delphi	C#
Name	String	public String
Part	array [0..7] of TOneTAC	public TOneTAC[] Part

TOneTAC Parametrik Yapısı		
	Delphi	C#
StartTime	TTime	public DateTime
EndTime	TTime	public DateTime

Fonksiyon çağırılmadan önce, cihazdan gelecek bilgilerin atanabilmesi için "***TTACList***" tipinde bir dizi değişken oluşturulmalıdır. Fonksiyon "***Boolean***" tipinde bir değer döndürmektedir. "***True***" bilgilerin başarıyla alındığını, "***false***" bilgilerin alınamadığı anlamına gelir. Programcı değerleri almadan önce 8 lik bir döngü oluşturmalı cihazdan gelen bilgileri "***TTACList***" dizi içersindeki "***StartTime***" ve "***EndTime***" elemanlarına atamalıdır.

Delphi

```
Var
TACList: TTACList;
i: Integer;
Rdr.GetTimeConstraintTables(integerDeğer, TACList);
for I := 0 to 7 do
begin
..... := TACList.Part[i].StartTime
..... := TACList.Part[i].EndTime;
end;
```

C#

```
TTACList TACList;
TACList = new TTACList();
rdr.GetTimeConstraintTables((byte)byteDeğer out TACList);
for (int i = 0; i <= 7; i++)
{
..... =TACList.Part[i].StartTime;
..... =TACList.Part[i].EndTime;
}
```

Zaman Kısıt Tablosu Gönderme İşlemi.

Zaman kısıt tablosu gönderme fonksiyonu "***SetTimeConstraintTables***" isimli fonksiyondur. Bu fonksiyon iki parametre ile çalışır. Birinci parametre "***byte***" tipinde olup "Tablo numarasını" temsil etmekte, ikinci parametre "***TTACList***" tipinde bir sınıf dizisidir. "***TTACList***" sınıf dizisi yukarıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
TACList: TTACList;
i: Integer;
TACList.Name := stringDeğer;
for I := 0 to 7 do
Begin
TACList.Part[i].StartTime := ......;
TACList.Part[i].EndTime := .....;
End;
Rdr.SetTimeConstraintTables(byteDeğer, TACList);
```

C#

```
TTACList TACList;
TACList = new TTACList();
TACList.Name = stringDeğer;
for (int i = 0; i <= 7; i++)
{
TACList.Part[i].StartTime =......;
TACList.Part[i].EndTime =......;
}
rdr.SetTimeConstraintTables((byte)byteVeri, TACList);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Anti Passback Ayarları.

Anti Passback sistemi 6 farklı modda çalışmaktadır.

1. Mod: Kapalı. Anti Passback etkin değil.
2. Mod: Anti Passback sistemi. Kartı okutan ve geçiş yapan kullanıcının, çıkış yapmadan yeniden giriş yapmasını (kart okutmasını) engelleyen bir kontrol mekanizmasıdır.
Aynı durum çıkış yapan kullanıcının, giriş yapmadan çıkış yapması için de geçerlidir.
3. Mod: Girişten Süre Kontrolü + Anti Passback. Kartı okutan ve geçiş yapan kullanıcının, iki geçiş arasındaki zaman tanımıdır. Bu adımda kullanıcı giriş yaptıktan sonra çıkış dahi yapsa girişe izin verilmez. İkinci geçiş için belirlenen sürenin dolması gerekmektedir. Ancak giriş yapan kullanıcı, belirtilen süre dışında da olsa çıkış yapabilir.
4. Mod: Süre Kontrolü + Anti Passback. Kartı okutan ve geçiş yapan kullanıcının, iki geçiş arasındaki zaman tanımıdır. Bu adımda kullanıcı giriş yaptıktan sonra çıkış dahi yapsa, girişe izin verilmez. İkinci geçiş için belirlenen sürenin dolması gerekmektedir. Kullanıcı bu mod da belirtilen süreler içerisinde çıkış dahi yapamaz.
5. Mod: Girişten Süre Kontrolü. Kullanıcı kartını okuttuktan sonra, belirlenen süre içerisinde tekrar girişten kart okutamaz ve giriş yapamaz.
6. Mod: Süre Kontrolü. Kullanıcının giriş ve çıkış sürelerini kontrol eder. Kullanıcı belirtilen süreler içerisinde giriş ve çıkış yapamaz.

TAPBSettings Parametrik Yapısı		
	Delphi	C#
APBType	byte	public byte
SequentialTransitionTime	byte	public byte
SecurityZone	byte	public byte
ApbInOut	byte	public byte

Anti Passback Ayarlarını Alma.

Anti PassBack ayarlarını alma fonksiyonu "***GetAntiPassbackSettings***" isimli fonksiyondur. Bu fonksiyon "***TAPBSettings***" tipinde bir sınıf dizisi parametre ile çalışmaktadır. Fonksiyon sadece "uygulama tipi" olarak "***fwPDKS***" tipinde çalışır. "***TAPBSettings***" sınıf dizisi ve elemanları aşağıda belirtilmiştir. Fonksiyondan dönen değer "***boolean***" tipindedir. "True" bilgilerin başarıyla alındığını "***false***" değeri bilgilerin alınamadığı anlamına gelir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Settings: TAPBSettings;
Rdr.GetAntiPassbackSettings(Settings);
..... := Settings.APBType;
..... := Settings.SequentialTransitionTime;
..... := Settings.SecurityZone;
..... := (Settings.ApbInOut = 0);
```

C#

```
TAPBSettings rSettings;
rSettings = new TAPBSettings();
rdr.GetAntiPassbackSettings(out rSettings);
..... = rSettings.APBType;
..... = rSettings.SequentialTransitionTime;
..... = rSettings.SecurityZone;
..... = (rSettings.ApbInOut == 0);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Anti Passback Ayarlarını Gönderme.

Anti ***PassBack*** ayarlarını gönderme fonksiyonu "***SetAntiPassbackSettings***" isimli fonksiyondur. Bu fonksiyon "***TAPBSettings***" tipinde bir sınıf dizisi parametre ile çalışmaktadır. Fonksiyon sadece "uygulama tipi" olarak "***fwPDKS***" tipinde çalışır. "***TAPBSettings***" sınıf dizisi ve elemanları yukarıda belirtilmiştir. Fonksiyondan dönen değer "***boolean***" tipindedir. "***True***" bilgilerin başarıyla alındığını "***false***" değeri bilgilerin alınamadığı anlamına gelir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Settings: TAPBSettings;
Settings.APBType := ....;
Settings.SequentialTransitionTime := .....;
Settings.SecurityZone := ....;
Settings.ApbInOut := 0; // yada 1
Rdr.SetAntiPassbackSettings(Settings);
```

C#

```
TAPBSettings rSettings;
rSettings = new TAPBSettings();
rSettings.APBType = (byte)....;
rSettings.SequentialTransitionTime = (byte)....;
rSettings.SecurityZone = (byte)....;
rSettings.ApbInOut = 0;// yada 1;
rdr.SetAntiPassbackSettings(rSettings);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Diğer EKS Ayarları / Alma.

Diğer EKS kişi bilgileri direkt karttan okunacaksa; yani herhangi bir pc desteği olmadan çalışacaksa bilgilerin yazılacağı sektörleri belirtir. Diğer EKS ayarlarını getiren fonksiyon "**GetEksOtherSettings**" isimli fonksiyondur. Bu fonksiyon "***TEksOtherSettings***" isimli bir sınıf dizisi parametresi ile çalışır. Cihazdan gelen bilgiler "***GetEksOtherSettings***" sınıf dizisinin elemanlarına atanır. "***TEksOtherSettings***" sınıf dizisi ve elemanları aşağıda belirtilmiştir. Bu fonksiyon sadece "***PDKS***" uygulama tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TEksOtherSettings Parametrik Yapısı		
	Delphi	C#
PersDataCardSector	byte	public byte
AccessDataCardSector	byte	public byte

Delphi

```
Var
EksOtherSettings: TEksOtherSettings;
Rdr.GetEksOtherSettings(EksOtherSettings);
.... := EksOtherSettings.PersDataCardSector;
.... := EksOtherSettings.AccessDataCardSector;
```

C#

```
TEksOtherSettings EksOtherSettings;
EksOtherSettings = new TEksOtherSettings();
rdr.GetEksOtherSettings(out EksOtherSettings);
..... = EksOtherSettings.PersDataCardSector;
..... = EksOtherSettings.AccessDataCardSector;
```

Diğer EKS Ayarları / Gönderme.

Diğer EKS ayarlarını gönderen fonksiyon "***SetEksOtherSettings***" isimli fonksiyondur. Bu fonksiyon "***TEksOtherSettings***" isimli bir sınıf dizisi parametresi ile çalışır. Cihaza gönderilecek bilgiler "***TEksOtherSettings***" sınıf dizisi tipinde bir değişkenin barındırdığı elemanlara atanır. "***TEksOtherSettings***" sınıf dizisi ve yukarıda belirtilmiştir. Bu fonksiyon sadece "***PDKS***" uygulama tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
EksOtherSettings: TEksOtherSettings;
EksOtherSettings.PersDataCardSector := .....;
EksOtherSettings.AccessDataCardSector := .....;
Rdr.SetEksOtherSettings(EksOtherSettings);
```

C#

```
TEksOtherSettings EksOtherSettings;
EksOtherSettings = new TEksOtherSettings();
EksOtherSettings.PersDataCardSector =(byte).....;
EksOtherSettings.AccessDataCardSector = (byte).....;
rdr.SetEksOtherSettings(EksOtherSettings);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Okuyucu Hizmet Dışı Ayarları.

Hizmet dışı okuyucu ayarları, belirlenen zaman dilimlerde kart okumayı kapatmak ve transistör çıkışlarının ne işlem yapacağını belirlemek amacıyla kullanılmaktadır. Hizmet dışı tablosunda 7 gün ve her güne 4 başlangıç ve 4 bitiş zamanı belirlemenize olanak tanır. Cihaz belirlediğiniz gün ve saatlerde devre dışı kalacak ve ayarladığınız transistörlerin durumları devreye girecektir. Hizmet dışı tablosu aşağıdaki dizilimler gibidir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

	Günler							
Günler	**1. Başlangıç**	**1. Bitiş**	**2. Başlangıç**	**2. Bitiş**	**3. Başlangıç**	**3. Bitiş**	**4. Başlangıç**	**4. Bitiş**
1.(pazartesi)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
2.(salı)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
3.(çarşamba)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
4.(perşembe)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
5.(cuma)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
6.(cumartesi)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
7.(pazar)	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd	ss.dd
ss.dd = saat.dakika								

Hizmet dışı tablosundan istenirse özel gün ve tatillerin listesi oluşturulabilir. Tatil listesine 30 adet veri girilebilir. Tatil listesi 2 değer ile çalışır. Tarih ve Haftanın günü değeri. Cihaz; belirlenen tarih ve haftanın günü numarasına karşılık gelen tarihte, haftanın günü numarasına karşılık gelen gündeymiş gibi davranarak işleme devam eder. Zaman dilimleri ayarlanırken, zamanların kesişmemesine dikkat edilmelidir.
Örneğin: Hizmet dışı tablosunda pazartesi gününe (1) karşılık gelen zaman dilimlerine aşağıdaki süreleri belirlemiş olalım.

1. Başlangıç 09:00 1. Bitiş 10:00
2. Başlangıç 11:00 2. Bitiş 12:00
3. Başlangıç 13:00 3. Bitiş 14:00
4. Başlangıç 15:00 4. Bitiş 16:00

Yukarıda belirtilen zaman dilimlerinde pazartesi günleri cihaz devre dışı kalacaktır. Yani herhangi bir kontrol mekanizması işletilmeyecektir. Zaman zaman, belirttiğiniz tatil ve özel günlerde cihazın çalışmamasını yada haftanın herhangi bir günü gibi davranmasını isteyebilirsiniz. Yani; 09.02.2015 tarihinde özel bir nedenden dolayı cihazın devre dışı kalmasını (herhangi bir kontrol mekanizması işletmemesini), ancak bu özel durumu haftanın herhangi bir gününde ayarladığınız gibi davranmasını isteyebilirsiniz. 09.02.2015 tarihinde (özel bir durum olduğu için) sanki pazar günündeymiş(7) gibi davran anlamına gelmektedir. Yukarıda belirtilenlere istinaden tatil listesi özel durum tablosu aşağıdakiler gibidir.

1. Tarih T.No (Gün No)
2. Tarih T.No (Gün No)
3. Tarih T.No (Gün No)
....

30.Tarih T.No (Gün No)

Hizmet dışı okuyucu ayarlarını getiren fonksiyon "***GetOutOfServiceTable***" isimli fonksiyondur. Bu fonksiyon "***TOSTable***" tipinde bir dizi parametresi ile çalışır. "***TOSTable***" dizisi ve elemanları aşağıda belirtilmiştir. Fonksiyona "***TOSTable***" tipinde parametre olarak gönderilmeden önce "***TOSTable***" tipinde bir dizi değişken oluşturulmalıdır. Cihazdan gelen bilgiler bu değişkenin içindeki elemanlara atanırlar.

TOneOOS Parametrik Yapısı		
	Delphi	C#
StartTime	TTime	public DateTime
EndTime	TTime	public DateTime

TDayOSS Parametrik Yapısı		
	Delphi	C#
part	array [0..3] of TOneOOS	public TOneOOS[] part;

TOSTable Parametrik Yapısı		
	Delphi	C#
Day	array [0..6] of TDayOSS	public TDayOSS[] day;

Delphi

```
Var
OSTable: TOSTable;
i, j: Integer;
frmMain.Rdr.GetOutOfServiceTable(OSTable);
for I := 0 to 6 do
for j := 0 to 3 do
Begin
..... := Copy(TimeToStr(OSTable.Day[i].part[j].StartTime), 1, 5);
..... :=Copy(TimeToStr(OSTable.Day[i].part[j].EndTime), 1, 5);
End;
```

C#

```
TOSTable OSTable = new TOSTable();
rdr.GetOutOfServiceTable(out OSTable);
for (int i = 0; i <= 6; i++)
{
for (int j = 0; j <= 3; j++)
{
..... = OSTable.day[i].part[j].StartTime);
..... = OSTable.day[i].part[j].EndTime);
}
}
```

Fonksiyondan geriye "***Boolean***" tipinde bir değer döner. Programcı isterse bu değeri kontrol ederek ayarların alınıp alınmadığı ile ilgili bilgi edinebilir.

Okuyucu Hizmet Dışı Ayarları / Gönderme.

Hizmet dışı okuyucu ayarlarını gönderen fonksiyon "***SetOutOfServiceSettings***" isimli fonksiyondur. Bu fonksiyon "***TOSTable***" tipinde bir dizi parametresi ile çalışır. "TOSTable" dizisi ve elemanları yukarıda belirtilmiştir. Fonksiyona "***TOSTable***" tipinde bir parametre göndermeden önce "***TOSTable***" dizisine ait elemanlara değerleri atanmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Settings: TOutOfServiceSettings;
Settings.Enabled := true; //ya da false
Settings.ScreenText1 := ......;
Settings.ScreenText2 := ......;
Settings.OutType := 1; // 0,1,2
Rdr.SetOutOfServiceSettings(Settings);
```

C#

```
TOutOfServiceSettings Settings;
Settings = new TOutOfServiceSettings();
Settings.Enabled = true; // ya da false
Settings.ScreenText1 = ......;
Settings.ScreenText2 = ......;
Settings.OutType = 0; // 0,1,2
rdr.SetOutOfServiceSettings(Settings);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Okuyucu Hizmet Dışı Tablosu Ayarları / Gönderme.

Hizmet dışı okuyucu ayarları "Hizmet dışı tablosunu" gönderen fonksiyon "***SetOutOfServiceTable***" isimli fonksiyondur. Bu fonksiyon "***TOSTable***" tipinde bir dizi parametresi ile çalışır. "***TOSTable***" dizisi ve elemanları yukarıda belirtilmiştir. Fonksiyona "***TOSTable***" tipinde bir parametre göndermeden önce bu dizi parametresinin elemanlarının değerlerinin atanması gerekir. "***TOSTable***" tipindeki dizinin elemanlarının tipleri konu başında belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
OSTable: TOSTable;
i, j: Integer;

for I := 0 to 6 do
 for j := 0 to 3 do
 Begin
 OSTable.Day[i].part[j].StartTime := .......;
 OSTable.Day[i].part[j].EndTime :=.......;
 End;
Rdr.SetOutOfServiceTable(OSTable);
```

C#

```
TOSTable OSTable = new TOSTable();
for (int i = 0; i < 6; i++)
{
for (int j = 0; j < 3; j++)
{
OSTable.day[i].part[j].StartTime = ......;
OSTable.day[i].part[j].EndTime = ........;
}
}
rdr.SetOutOfServiceTable(OSTable);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Tatil Listesi / Alma.

Tatil Listesini alan fonksiyon "***GetOutOfServiceHolidayList***" isimli fonksiyondur. Bu fonksiyon "***THolidays***" tipinde bir dizi parametresi ile çalışır. "***THolidays***" tipindeki dizi değişkenler ve elemanları aşağıda belirtilmiştir. "***GetOutOfServiceHolidayList***" fonksiyonu çağırılmadan önce "***THolidays***" sınıf dizisi tipinde bir değişken oluşturulmalıdır. Cihazdan gelen bilgiler bu değişkenin elemanlarına atanmakdatır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

THolidayDate Parametrik Yapısı		
	Delphi	C#
Date	TDate	public DateTime
OOSTableNo	Byte	public Byte

THolidays Parametrik Yapısı		
	Delphi	C#
List	array [0..29] of THolidayDate	public THolidayDate[] List = new THolidayDate[29];

Delphi

```
Var
Holidays: THolidays;
i: Integer;
Rdr.GetOutOfServiceHolidayList(Holidays);
for I := 0 to 29 do
Begin
..... := DateToStr(Holidays.List[i].Date);
..... := IntToStr(Holidays.List[i].OOSTableNo);
End;
```

C#

```
THolidays Holidays;
rdr.GetOutOfServiceHolidayList(out Holidays);
for (int i = 0; i <=29 ; i++)
{
..... = Holidays.List[i].Date;
..... =Convert.ToString(Holidays.List[i].OOSTableNo);
}
```

Parametre Anlamları
Tatil listesi; hizmet dışı tablosu ile entegre çalışır. Herhangi bir tatil listesi kaydı oluşturmadan önce, hizmet dışı tablosunun yapılandırılması, yani en az bir kayıt oluşturulması gerekir. Tatil listesi oluşturmakta ki amaç; herhangi özel gün veya tatillerde, girilen tarihin, hizmet dışı tablosundaki gibi davranmasını sağlamaktır. Örneğin 01.01.2015 günü Perşembe gününe gelmektedir ve resmi tatildir. Ancak belli saatler arasında resmi tatil değilmiş gibi ya da tersi gibi davranmak istenebilir. Bu durumda tatil listesi ayarlanmalıdır. ***List:*** *Tatil listelerini temsil eder. Bu ayarı yapmak için bir tarih belirtilmeli ve hizmet dışı tablosunda karşılık gelen saat aralıkları numarası verilmelidir.*

Tatil Listesi / Alma.

Tatil Listesini alan fonksiyon "***GetOutOfServiceHolidayList***" isimli fonksiyondur. Bu fonksiyon "***THolidays***" tipinde bir dizi parametresi ile çalışır. "***THolidays***" tipindeki dizi değişkenler ve elemanları yukarıda belirtilmiştir. "***GetOutOfServiceHolidayList***" fonksiyonu çağırılmadan önce "***THolidays***" sınıf dizisi tipinde bir değişken oluşturulmalıdır. Bu değişkenin elemanlarına değerleri atanmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Holidays: THolidays;
i: Integer;
for I := 0 to 29 do
Begin
Holidays.List[i].Date := .....;
Holidays.List[i].OOSTableNo := .....;
End;

Rdr.SetOutOfServiceHolidayList(Holidays);
```

C#

```
THolidays Holidays;
Holidays = new THolidays();
for (int i = 0; i <= 29; i++)
{
Holidays.List[i].Date = ......;
Holidays.List[i].OOSTableNo = ......;
}
rdr.SetOutOfServiceHolidayList(Holidays);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Tatil Listesi / Gönderme.

Tatil Listesini gönderen fonksiyon "***SetOutOfServiceHolidayList***" isimli fonksiyondur. Bu fonksiyon "***THolidays***" tipinde bir dizi parametresi ile çalışır. "***THolidays***" tipindeki dizi değişkenler ve elemanları yukarıda belirtilmiştir."***SetOutOfServiceHolidayList***" fonksiyonu çağırılmadan önce "***THolidays***" sınıf dizisi tipinde bir değişken oluşturulmalıdır. Bu değişkenin elemanlarına değerleri atanmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Holidays: THolidays;
i: Integer;
for I := 0 to 29 do
Begin
Holidays.List[i].Date := .....;
Holidays.List[i].OOSTableNo := .....;
End;
Rdr.SetOutOfServiceHolidayList(Holidays);
```

C#

```
THolidays Holidays;
Holidays = new THolidays();
for (int i = 0; i <= 29; i++)
{
Holidays.List[i].Date = ......;
Holidays.List[i].OOSTableNo = ......;
}
rdr.SetOutOfServiceHolidayList(Holidays);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Zil Tablosu Ayarları Genel Ayarları / Alma.

Mevcut cihaz ile zil çaldırma işlemi de yapılabilmektedir. Zil çalarken aynı zamanda ekranın 1. ve 2. satırlarında mesaj yayınlanabilir. Bunun için "zil çaldırma etkin" seçili hale getirilmelidir. Zil çaldırma fonksiyonu "Zil tablosu" ayarları ile birlikte çalışır. Zil Tablosu 0 dan 6 ya kadar, günleri temsil eden değerler alır. 0 pazartesi 6 pazar olarak belirlenmiştir. Her güne toplam 24 adet zil zamanı ataması yapabilir ve zil çalma sürelerini ayarlayabilirsiniz.
Zil tablosu aşağıdaki yapıda olmalıdır.
Gün 0..6 (0 Pazartesi. . . . . 6 Pazar)

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Zil zaman tablosu.		
Zil	**Saat**	**Çalma Süresi**
1	ss.dd	saniyenin 10/1
2	ss.dd	saniyenin 10/1
.	ss.dd	saniyenin 10/1
.	ss.dd	saniyenin 10/1
24	ss.dd	saniyenin 10/1

Bu fonksiyon sadece "***PDKS***" uygulama tipinde çalışır. Zil Tablosu Genel Ayarlarını getiren fonksiyon "***GetBellSettings***" fonksiyonudur. Bu fonksiyon "***TBellSettings***" tipinde bir dizi parametresi ile çalışır. "***TBellSettings***" sınıf dizisi ve elamanlarını aşağıda bulabilirsiniz.

TBellSettings parametrik yapıları		
	Delphi	**C#**
Enabled	Boolean	public Boolean
ScreenText1	string	public string
ScreenText2	string	public string
OutType	Byte	public Byte

Delphi

```
Var
Settings: TBellSettings;
Rdr.GetBellSettings(Settings);
.... := Settings.Enabled; // false ya da true değeri döner
.... := Settings.ScreenText1;
.... := Settings.ScreenText2;
.... := Settings.OutType; // 1 ya da 2 değeri döner
```

C#

```
TBellSettings Settings;
Settings = new TBellSettings();
dr.GetBellSettings(out Settings);
.... = Settings.Enabled; //false ya da true değeri döner
.... = Settings.ScreenText1;
.... = Settings.ScreenText2;
.... = Settings.OutType; // 1 ya da 2 değeri döner.
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Zil Tablosu Genel Ayarları / Gönderme.

Zil çaldırma genel ayarlarını gönderme fonksiyonu "***SetBellSettings***" fonksiyonudur. Bu fonksiyon "***TBellSettings***" tipinde bir dizi parametresi ile çalışır. "***TBellSettings***" dizisinin elemanları yukarıda belirtilmiştir. "***TBellSettings***" dizi elemanlarına, değerleri atanmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Settings: TBellSettings;
Settings.Enabled :=true ;// ya da false;
Settings.ScreenText1 := .....;
Settings.ScreenText2 := .....;
Settings.OutType := 1; // ya da 2
Rdr.SetBellSettings(Settings);
```

C#

```
TBellSettings Settings;
Settings = new TBellSettings();
Settings.Enabled = true;// ya da false
Settings.ScreenText1 = ......;
Settings.ScreenText2 = ......;
Settings.OutType = 1; // ya da 2
rdr.SetBellSettings(Settings);
```

Programcı isterse fonksiyondan geri dönen "***boolean***" tipinden değeri kontrol ederek bilgilerin alınıp alınmadığı hakkında bilgi edinebilir.

Zil Tablosu / Alma.

Zil tablosunu alma fonksiyonu "***GetBellTable***" isimli fonksiyondur. Bu fonksiyon "***Byte***" tipinde ve "***TBellTable***" tipinde bir dizi parametresi ile çalışır. "***Byte***" tipindeki parametre günü temsil eder. Fonksiyonun genel çalışma prensibi *GetBellTable(byteDeğer,TBellTableTipindekiDeğer)* şeklindedir. "***TBellTable***" dizisi ve elemanları aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TOneBell Parametrik Yapıları		
	Delphi	C#
StartTime	TTime	public DateTime
Duration	byte	public byte

TBellTable Parametrik Yapıları		
	Delphi	C#
List	array [0..23] of TOneBell	public TOneBell[] List *public TBellTable()* *{List = new TOneBell[23];}*

Delphi

```
Var
BellTable: TBellTable;
Rdr.GetBellTable(byteDeğer, BellTable);
for I := 0 to 23 do
Begin
..... :=BellTable.List[i].StartTime);
..... :=BellTable.List[i].Duration);
End;
```

C#

```
TBellTable BellTable;
rdr.GetBellTable((byte)bytedeğer, out BellTable);
for (int i = 0; i <= 23; i++
{
BellTable.List[i].StartTime;
}
```

Zil Tablosu / Gönderme.

Zil tablosu gönderme fonksiyonu "***SetBellTable***" isimli fonksiyondur. Bu fonksiyon "***Byte***" tipinde ve "***TBellTable***" tipinde bir dizi parametresi ile çalışır. "***Byte***" tipindeki parametre günü temsil eder. Cihaza parametreler gönderilmeden önce "***TBellTable***" tipinde bir değişken oluşturulmalı ve bu dizi değişkeninin elemanlarına, değerleri atanmalıdır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
BellTable: TBellTable;
i: Integer;
for I := 0 to 23 do
Begin
BellTable.List[i].StartTime:=.....;
BellTable.List[i].Duration := .....;
End;
```

C#

```
TBellTable BellTable;
BellTable = new TBellTable();
for (int i = 0; i <= 23; i++)
{
BellTable.List[i].StartTime = ........;
BellTable.List[i].Duration  = ........;
}
rdr.SetBellTable((byte)txtGun.Value, BellTable);
```

Zil Tablosu / Fabrika Ayarlarına Dönme.

Zil tablosu ayarlarını fabrika ayarlarına getiren fonksiyon "***SetAppFactoryDefault***" isimli fonksiyondur. Bu fonksiyon parametre olarak "***boolean***" tipinde bir değer almaktadır. "True" değer gönderilirse cihaz fabrika ayarlarına döner ve yeniden başlatılır. Fabrika ayarlarına dönme komutu çalıştırıldıktan sonra cihazla olan bağlantı kesilmeli ve bağlantı öncesi kısa bir süre beklenilmelidir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
rdr.SetAppFactoryDefault(true); // ya da false. True reboot edilsin false reboot edilmesin anlamına gelir.
Rdr.DisConnect;
Sleep(100);
Rdr.Connect;
Rdr.SetBellTable(byteDeğer, BellTable);
```

Delphi

```
rdr.SetAppFactoryDefault(true);
rdr.DisConnect();
Thread.Sleep(1000);
rdr.Connect();
```

Tanımlı Tüm Kişileri Silme.

Tanımlı Tüm Kişileri Silme fonksiyonu "***ClearWhitelist***" fonksiyonudur. Bu fonksiyon parametre almadan çalışır. "rdr" nesnesinin "***ClearWhitelist***" olayının tetiklenmesi ardından işlem gerçekleşir. Bu fonksiyon sadece uygulama tipi olarak "***PDKS***" ve "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.ClearWhitelist;
```

Delphi

```
rdr.ClearWhitelist();
```

Tanımlı Kişi Ekleme.

Kişi ekleme fonksiyonu "***AddWhitelist***" isimli fonksiyondur. Bu fonksiyon "***TPerson***" ve "***integer***" tipinde iki parametre ile çalışır. Bu fonksiyon sadece uygulama tipi olarak "***PDKS***" tipinde çalışır. "***TPerson***" isimli bir sınıf dizi yapısıdır. Fonksiyona parametre olarak gönderilmeden önce sınıf dizisi oluşturulmalı ve elemanlarına değerleri atanmalıdır. "***TPerson***" tipindeki sınıf dizi aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TPerson Parametrik Yapıları		
CardID	String	public string
Name	String	public string
TimeAccessMask	TimeAccessMask	public byte[] this.TimeAccessMask = new byte[7];
Code	Code	public uint
Password	Password	public ushort
EndDate	EndDate	public byte[] this.EndDate = new byte[3];
AccessDevice	AccessDevice	public Boolean
APBEnabled	APBEnabled	public Boolean
ATCEnabled	ATCEnabled	public Boolean
AccessCardEnabled	AccessCardEnabled	public Boolean

Dizi elemanları içinde yer alan "CardID" elemanının uzunluğu 14 karakter ve Hexadecimal sayı olmak durumundadır. Bu nedenle parametre olarak gönderilmeden önce kontrol edilmesinde yarar olacaktır. Eğer gelen "CardID" değeri 14 karakterden küçük ise Sağdan "0" (sıfır) ile boşlukları doldurmak gerekir. Dizi elemanları içinde yer alan "Name" elemanın tipi string ve en fazla 18 karakter olmak zorundadır. Dizi elemanları içinde yer alan "TimeAccessMask"; 0 pazartesi, 7 pazar olacak şekildi günleri temsil eder. Yine "TimeAccessMask[x]" dizi elemanları içindeki "x" zaman kısıt tablosunda yer alan tabloları ifade etmektedir. Zaman kısıt tablosuna daha önceki konurlarda değinilmiştir.

Fonksiyondan geri 0,1,20,51,52 değerleri dönmektedir.
0: Kişi eklendi.
1: Kişi daha önce eklenmiş.
20: Şifre kapasitesi aşıldı.
51: Kart Id daha önce tanımlanmış.
52: Şifre daha önce tanımlanmış.

Parametre Anlamları
CardID: *Kullanıcıya tanımlanacak cartın ID sini temsil eder.* ***Name:*** *Tanımlanacak kullanıcının adını temsil eder.* ***TimeAccessMask:*** *Haftanın 7 gününe denk gelen bir tanımı temsil eder. 0 pazartesi ….. 7 Pazar anlamına gelir. Kullanıcı bu tarihlerde kartını okutamaz. Kart okuyucu işlem yapmaz.* ***Code:*** *Kullanıcıya tanımlanan kullanıcı kodunu temsil eder.* ***Password:*** *Kullanıcıya tanımlanan kullanıcı şifresini temsil eder.* ***EndDate:*** *Kullanıcıya tanımlanan son kullanma tarihini temsil eder. Bu tarihten sonra kullanıcı kartını kullanamaz.* ***AccessDevice:*** *Kullanıcının sistemde aktif olmup olmadığını temsil eder.* ***APBEnabled:*** *Kullanıcı için anti pass back (çıkış yapmadan giremez, giriş yapmadan çıkış yapamaz.)* ***ATCEnabled:*** *Kullanıcı için zaman kısıt tablosu aktif olup olmama durumunu temsil eder. True olması durumunda cihaz, kullanıcı zaman kısıt tablosundaki bilgilere göre aksiyon alacakdır.* ***AccessCardEnabled:*** *Giriş kartının sistemde aktif olup olmadığını temsil eder.*

Delphi

```
var
Person: TPerson;
InxNm: Integer;
CardID: String;

Person.CardID := CardID;
Person.Name := ......;
Person.TimeAccessMask[0] := ....;
Person.TimeAccessMask[1] := ....;
Person.TimeAccessMask[2] := ....;
Person.TimeAccessMask[3] := ....;
Person.TimeAccessMask[4] := ....;
Person.TimeAccessMask[5] := ....;
Person.TimeAccessMask[6] := ....;
Person.Code := .....;
Person.Password := ....; // Integer
Person.EndDate := .....;
Person.AccessDevice := true//ya da false;
Person.APBEnabled := true//ya da false;
Person.ATCEnabled := true//ya da false;
Person.AccessCardEnabled := true//ya da false;
Rdr.AddWhitelist(Person, InxNm);
```

C#

```
int iErr;

uint InxNm;

string CardID;
TPerson Person = new TPerson();
Person.CardID = CardID;
Person.Name = .....;
Person.TimeAccessMask[0] = (byte).....;
Person.TimeAccessMask[1] = (byte).....;
Person.TimeAccessMask[2] = (byte).....;
Person.TimeAccessMask[3] = (byte).....;
Person.TimeAccessMask[4] = (byte).....;
Person.TimeAccessMask[5] = (byte).....;
Person.TimeAccessMask[6] = (byte).....;
Person.Code = (uint).....;
Person.Password = (ushort).....;// ushort
Person.EndDate[0] = (byte).....;
Person.EndDate[1] = (byte).....;
Person.EndDate[2] = (byte).....;
Person.AccessDevice = true//ya da false;
Person.APBEnabled = true//ya da false;
Person.ATCEnabled = true//ya da false;
Person.AccessCardEnabled = true//ya da false;
rdr.AddWhitelist(Person,out InxNm);
```

Tanımlı Kişi Değiştirme.

Kişi Değiştirme fonksiyonu "***EditWhitelistWithCardID***" isimli fonksiyondur. Bu fonksiyon sadece uygulama tipi olarak "***PDKS***" tipinde çalışır. "***TPerson***" isimli bir sınıf dizi yapısıdır. Fonksiyona parametre olarak gönderilmeden önce sınıf dizisi oluşturulmalı ve elemanlarına değerleri atanmalıdır. "***TPerson***" tipindeki sınıf dizi yukarıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
Person: TPerson;
InxNm: Integer;
CardID: String;
Person.CardID := CardID;
Person.Name := ......;
Person.TimeAccessMask[0] := ....;
Person.TimeAccessMask[1] := ....;
Person.TimeAccessMask[2] := ....;
Person.TimeAccessMask[3] := ....;
Person.TimeAccessMask[4] := ....;
Person.TimeAccessMask[5] := ....;
Person.TimeAccessMask[6] := ....;
Person.Code := .....;
Person.Password := ....; // Integer
Person.EndDate := .....;
Person.AccessDevice := true//ya da false;
Person.APBEnabled := true//ya da false;
Person.ATCEnabled := true//ya da false;
Person.AccessCardEnabled := true//ya da false;
Rdr.EditWhitelistWithCardID(Person, InxNm);
```

C#

```
int iErr;
uint InxNm;
string CardID;
TPerson Person = new TPerson();
Person.CardID = CardID;
Person.Name = .....;
Person.TimeAccessMask[0] = (byte).....;
Person.TimeAccessMask[1] = (byte).....;
Person.TimeAccessMask[2] = (byte).....;
Person.TimeAccessMask[3] = (byte).....;
Person.TimeAccessMask[4] = (byte).....;
Person.TimeAccessMask[5] = (byte).....;
Person.TimeAccessMask[6] = (byte).....;
Person.Code = (uint).....;
Person.Password = (ushort).....;// ushort
Person.EndDate[0] = (byte).....;
Person.EndDate[1] = (byte).....;
Person.EndDate[2] = (byte).....;
Person.AccessDevice = true//ya da false;
Person.APBEnabled = true//ya da false;
Person.ATCEnabled = true//ya da false;
Person.AccessCardEnabled = true//ya da false;
rdr.EditWhitelistWithCardID(Person,out InxNm);
```

Tanımlı Kişi Silme.

Kişi silme fonksiyonu "***DeleteWhitelistWithCardID***" isimli fonksiyondur. Bu fonksiyon "***CardID***" temsilen "***String***" ve "***InxNm***" temsilen "Integer" tipinde iki parametre ile çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.DeleteWhitelistWithCardID(CardID, InxNm);
```

C#

```
rdr.DeleteWhitelistWithCardID(CardID, out InxNm);
```

Tanımlı Kişi Bulma.

Kişi bulma fonksiyonu "***GetWhitelistWithCardID***" isimli fonksiyondur. Bu fonksiyon 3 parametre ile çalışır. Birinci parametre "***String***" tipinde, ikinci parametre "***TPerson***" tipinde, üçüncü parametre "***Integer***" tipinde parametredir.
Genel kullanımı *Rdr.GetWhitelistWithCardID(stringDeğer, TPersonDeğer, Integerdeğer)* şeklindedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
Person: TPerson;
iErr, i, cnt, InxNm: Integer;
CardID: String;
Rdr.GetWhitelistWithCardID(CardID, Person, InxNm);
..... := Person.TimeAccessMask[0];
..... := Person.TimeAccessMask[1];
..... := Person.TimeAccessMask[2];
..... := Person.TimeAccessMask[3];
..... := Person.TimeAccessMask[4];
..... := Person.TimeAccessMask[5];
..... := Person.TimeAccessMask[6];
..... := Person.Code;
..... := Person.Password;
..... := Person.EndDate;
..... := Person.AccessDevice;
..... := Person.APBEnabled;
..... := Person.ATCEnabled;
..... := Person.AccessCardEnabled;
..... := IntToStr(InxNm);
```

C#

```
int iErr;
uint InxNm;
string CardID;
TPerson Person = new TPerson();
rdr.GetWhitelistWithCardID(CardID,out Person,out InxNm);
.... = Person.TimeAccessMask[0];
.... = Person.TimeAccessMask[1];
.... = Person.TimeAccessMask[2];
.... = Person.TimeAccessMask[3];
.... = Person.TimeAccessMask[4];
.... = Person.TimeAccessMask[5];
.... = Person.TimeAccessMask[6];
.... = Person.Code;
.... = Person.Password;
.... = new DateTime(Person.EndDate[2] + 2000, Person.EndDate[1], Person.EndDate[0], 0, 0, 0);
.... = Person.AccessDevice;
.... = Person.APBEnabled;
.... = Person.ATCEnabled;
.... = Person.AccessCardEnabled;
.... = InxNm.ToString();
```

Tanımlı Kişi Sayısı Bulma.

Cihazda tanımlı kişilerin sayısını alma işlemini gerçekleştiren fonksiyon "***GetWhitelistCardIDCount***" isimli fonksiyondur. Bu fonksiyon parametresiz olarak çalışmaktadır. ***Rdr*** nesnesinin "***GetWhitelistCardIDCount***" isimli olayı tetiklenmesi ile çalışır. Fonksiyondan geriye "Integer" tipinde değer döner. Bu fonksiyon sadece uygulama tipi olarak "***PDKS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
.....:=Rdr.GetWhitelistCardIDCount;
```

C#

```
..... =rdr.GetWhitelistCardIDCount();
```

PDKS Cihazdan Giriş/Çıkış Bilgilerini Transfer Etme.

Program içinden cihaza giriş çıkış bilgileri transfer etmek için kullanılan fonksiyondur. Bu işlemi gerçekleştiren fonksiyon "***TransferRecords***" isimli fonksiyondur. Fonksiyon "***TAccessRecords***" tipinde bir parametre ile çalışır. ***"TAccessRecords"*** sınıf dizisi ve eleman tipleri aşağıda belirtilmiştir. Bu fonksiyon sadece uygulama tipi olarak "***PDKS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

TAccessRecords Parametrik Yapıları		
	Delphi	C#
Count	Cardinal	public uint Count;
raDeviceRecs	array of TOneRecord	public TOneRecord[] raDeviceRecs

TOneRecord Parametrik Yapıları		
	Delphi	C#
CardID	string	public string CardID
DoorNo	Byte	public byte DoorNo
RecordType	Byte	public byte RecordType
TimeDate	TDateTime	public DateTime TimeDate

Delphi

```
var
tempRecs: TAccessRecords;
i: Integer;
strCardId: String;
Rdr.TransferRecords(tempRecs);
for i := 0 to tempRecs.Count - 1 do
begin
strCardId := tempRecs.raDeviceRecs[i].CardID;
mmFile.Lines.Add('Card ID: ' + strCardId + ' ' + 'Door No: ' +
IntToStr(tempRecs.raDeviceRecs[i].DoorNo) + ' ' + 'Record Type: ' +
IntToStr(tempRecs.raDeviceRecs[i].RecordType) + ' ' + 'RFU[0]: ' +
IntToStr(tempRecs.raDeviceRecs[i].RFU[0]) + ' ' + 'RFU[1]: ' +
IntToStr(tempRecs.raDeviceRecs[i].RFU[1]) + ' ' + 'Time Date : ' +
DateTimeToStr(tempRecs.raDeviceRecs[i].TimeDate));
end;
```

C#

```
TAccessRecords tmpRecs = new TAccessRecords();
rdr.TransferRecords(out tmpRecs);
for (int i = 0; i < tmpRecs.Count; i++)
{
…. = tmpRecs.raDeviceRecs[i].CardID.ToString();
…. = tmpRecs.raDeviceRecs[i].TimeDate.ToString();
…. = tmpRecs.raDeviceRecs[i].DoorNo.ToString();
…. = tmpRecs.raDeviceRecs[i].RecordType.ToString();
}
```

PDKS Cihazdan Giriş/Çıkış Bilgilerini Getirme.

Program içinden cihaza giriş çıkış bilgileri transfer etmek için kullanılan fonksiyondur. Bu işlemi gerçekleştiren fonksiyon "***TransferRecords***" isimli fonksiyondur. Fonksiyon "***TAccessRecords***" tipinde bir parametre ile çalışır. "***TAccessRecords"*** sınıf dizisi ve eleman tipleri yukarıda belirtilmiştir. Fonksiyon StartFrom,HowMany:***Cardinal*** ve out Recs ***TAccessRecords*** tiplerinde 3 parametre ile çalışır. Fonksiyondan dönen "***boolean***" türündedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
tempRecs: TAccessRecords;
i: Integer;
strCardId: String;
Rdr.ReadRecords(seStartFrom.Value, seHowMany.Value, tempRecs);
for i := 0 to tempRecs.Count - 1 do
begin
strCardId := tempRecs.raDeviceRecs[i].CardID;
…. :=IntToStr(tempRecs.raDeviceRecs[i].DoorNo) ;
…. := IntToStr(tempRecs.raDeviceRecs[i].RecordType) ;
…. := IntToStr(tempRecs.raDeviceRecs[i].RFU[0]);
…. := IntToStr(tempRecs.raDeviceRecs[i].RFU[1]);
…. := DateTimeToStr(tempRecs.raDeviceRecs[i].TimeDate));
end;
```

C#

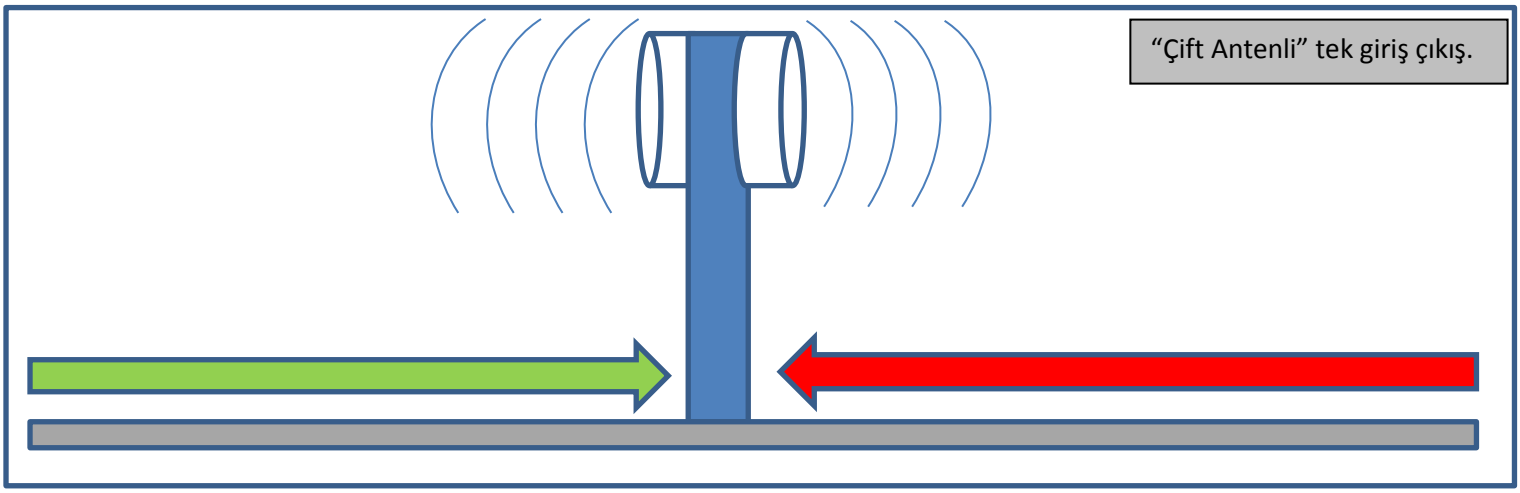
```
TAccessRecords tempRecs = new TAccessRecords();
uint Start = (uint)edtStartRead.Value;
uint HowMany = (uint)edtHowManyRead.Value;
rdr.ReadRecords(Start,HowMany,out tempRecs);
for (int i = 0; i < tempRecs.Count; i++)
{
…. = tempRecs.raDeviceRecs[i].CardID.ToString()
…. = tempRecs.raDeviceRecs[i].TimeDate.ToString()
…. = tempRecs.raDeviceRecs[i].DoorNo.ToString()
…. = tempRecs.raDeviceRecs[i].RecordType.ToString();
}
```

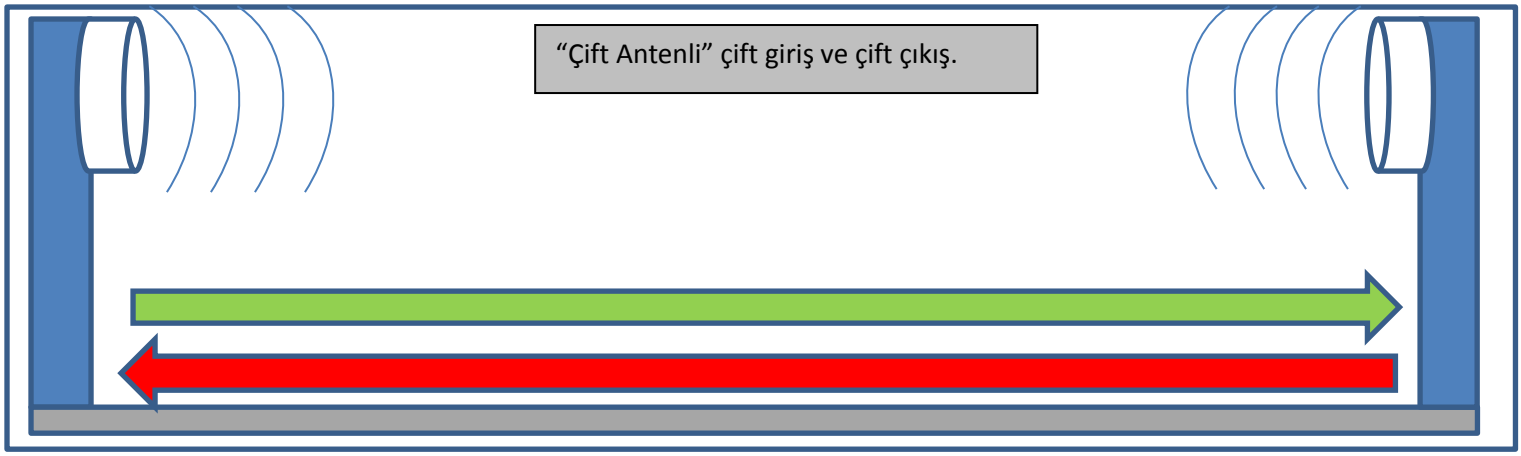
HGS Genel Ayarları / Getirme.

HGS Genel ayarlarını alma fonksiyonu " ***GetHGSSettings***" fonksiyonudur. Bu fonksiyon " "***THGS_Settings***" tipinde bir sınıf dizisi parametresi alır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "***True***" değeri bilgilerin cihazdan başarıyla alındığını "***False***" değeri cihazdan bilgilerin getirilemediği anlamına gelir. "***Settings"*** sınıf dizisinin parametrik yapıları aşağıda belirtilmiştir. "***ProgramMode***" isimli parametre HGS cihazının çalışma modunu temsil eder.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

ProgramMode Parametrik Yapıları	
	Açıklama
Tek Anten	*Girişlerde, tek anten kullanılır. Giriş bilgisi ve çıkış bilgisi aynı kapıdan ve aynı antenden alınır.*
Çift Anten Tek Geçiş	*Girişlerde ve çıkışlarda, çift anten kullanılır. Girişler ve çıkışlar aynı kapıdan yapılır fakat bilgi iletimi ayrı iki antenden alınır.*
Çift Anten Çift Geçiş	*Girişlerde ve çıkışlarda, çift anten kullanılır. Girişler ve çıkışlar ayrı kapıdan yapılır ve bilgi iletimi ayrı iki antenden yapılır.*





THGS_Settings Parametrik Yapıları		
	Delphi	C#
PaketBoyu	byte	Public byte
CardBaslangic	Byte	Public Byte
CardBoyu	Byte	Public Byte
TrCikisSure1	Byte	Public Byte
TrCikisSure2	Byte	Public Byte
ProgramMode	Byte	Public Byte
DiziEklemeSure1	Byte	Public Byte
DiziEklemeSure2	Byte	Public Byte
ZamanKisitEnb	Boolean	Public Boolean
AntenPower1	byte	Public Byte
AntenPower2	byte	Public byte
AntenTanitim	byte	Public byte
DefMaksimumArac	byte	Public byte
DefAntiPassPack	byte	Public byte

Delphi

```
var
rSettings: THGS_Settings;
Rdr.GetHGSSettings(rSettings);
….. := rSettings.PaketBoyu;
….. := rSettings.CardBaslangic;
….. := rSettings.CardBoyu;
….. := rSettings.TrCikisSure1;
….. := rSettings.TrCikisSure2;
….. :=  rSettings.ProgramMode-1;
….. := rSettings.DiziEklemeSure1;
….. := rSettings.DiziEklemeSure2;
….. := rSettings.AntenPower1;
….. := rSettings.AntenPower2;
….. := rSettings.AntenTanitim;
….. := rSettings.DefMaksimumArac;
….. := rSettings.DefAntiPassPack;
```

C#

```
var
THGS_Settings rSettings;
rSettings = new THGS_Settings();
rdr.GetHGSSettings(out rSettings);
…… = rSettings.PaketBoyu;
…… = rSettings.CardBaslangic;
…… = rSettings.CardBoyu;
…… = rSettings.TrCikisSure1;
…… = rSettings.TrCikisSure2;
…… = rSettings.ProgramMode -1;
…… = rSettings.DiziEklemeSure1;
…… = rSettings.DiziEklemeSure2;
```

HGS Genel Ayarları / Gönderme.

HGS Genel ayarlarını gönderme fonksiyonu "***SetHGSSettings***" fonksiyonudur. Bu fonksiyon " "***THGS_Settings***" tipinde bir sınıf dizisi parametresi alır. Fonksiyon "***Boolean***" tipinde bir değer döndürür. "***True***" değeri bilgilerin cihazdan başarıyla alındığını "***False***" değeri cihazdan bilgilerin getirilemediği anlamına gelir. "***Settings"*** sınıf dizisinin parametrik yapıları aşağıda belirtilmiştir. "***ProgramMode***" isimli parametre HGS cihazının çalışma modunu temsil eder. "***THGS_Settings"*** parametresi gönderilmeden önce diziye ait elemanların değerlerinin atanması gerekmektedir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
rSettings: THGS_Settings;
rSettings.PaketBoyu := edtHGSPaketBoyu.Value;
rSettings.CardBaslangic := edtHGSKartBaslangic.Value ;
rSettings.CardBoyu := edtHGSKartBoyu.Value;
rSettings.TrCikisSure1 := edtHGSRLOut1.Value;
rSettings.TrCikisSure2 := edtHGSRLOut2.Value;
rSettings.ProgramMode := cbProgramModu.ItemIndex+1;
rSettings.DiziEklemeSure1 := edtHGSDiziAyar1.Value;
rSettings.DiziEklemeSure2 := edtHGSDiziAyar2.Value;
rSettings.AntenPower1 := edtHGSAntenGuc1.Value;
rSettings.AntenPower2 := edtHGSAntenGuc2.Value;
rSettings.AntenTanitim := edtHGSAntenGucKTanima.Value;
rSettings.DefMaksimumArac := edtHGSVDaireAS.Value;
rSettings.DefAntiPassPack := edtHGSApb.Value;
Rdr.SetHGSSettings(rSettings);
```

C#

```
THGS_Settings rSettings;
rSettings = new THGS_Settings();
rSettings.PaketBoyu = (byte)txtHgsGenelAyarlarSeriPaketBoyutu.Value;
rSettings.CardBaslangic = (byte)txtHgsGenelAyarlarKartBaslangici.Value;
rSettings.CardBoyu = (byte)txtHgsGenelAyarlarTagId.Value;
rSettings.TrCikisSure1 = (byte)txtHgsGenelAyarlarTransistorCikisi1.Value;
rSettings.TrCikisSure2 = (byte)txtHgsGenelAyarlarTransistorCikisi2.Value ;
rSettings.ProgramMode =  (byte)(txtHgsGenelAyarlarProgramModu.SelectedIndex +1) ;
rSettings.DiziEklemeSure1 = (byte)txtHgsGenelAyarlarDiziArdisikKontrol1.Value;
rSettings.DiziEklemeSure2 = (byte)txtHgsGenelAyarlarDiziArdisikKontrol2.Value;
rSettings.ZamanKisitEnb =true; // yada false
rSettings.AntenPower1 = (byte)txtHgsGenelAyarlarAntenGuc1.Value;
rSettings.AntenPower2 = (byte)txtHgsGenelAyarlarAntenGuc2.Value;
rSettings.AntenTanitim = (byte)txtHgsGenelAyarlarAntenKtanima.Value;
rSettings.DefMaksimumArac = (byte)txtHgsGenelAyarlarAracDaireSayisi.Value;
rSettings.DefAntiPassPack = (byte)txtHgsGenelAyarlarArdisikGecisSuresi.Value;
rdr.SetHGSSettings(rSettings);
```

HGS’de Tanımlı Tüm Kişileri Silmek.

HGS’de tanımlı tüm kişileri silme işlemi fonksiyonu "***ClearWhitelist***" fonksiyonudur. Bu fonksiyon parametresiz olarak çalışır. Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.ClearWhitelist;
Rdr.DisConnect;
Sleep(100);
Rdr.Connect;
```

HGS’de TAG Ekleme İşlemi.

HGS sistemine TAG ekleme işlemi “***AddHGSWhitelist***” isimli fonksiyondur. Bu fonksiyon “***THGSArac***” ve “***Integer***” tipinde iki parametre ile çalışır. Fonksiyonun genel kullanımı *Rdr.AddHGSWhitelist(Arac,InxNm); “**THGSArac”*** sınıf dizisinin parametrik yapıları aşağıda belirtilmiştir. Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

THGSArac Parametrik Yapıları		
	Delphi	C#
CardID	string	Public string
Name	string	Public string
TimeAccessMask	array [0..6] of byte	Public byte[]
Daire	Word	Public ushort
DaireHak	byte	Public byte
AracNo	byte	Public byte
EndDate	Tdate	Public DateTime
AccessDevice	Boolean	Public Boolean
APBEnabled	Boolean	Public Boolean
ATCEnabled	Boolean	Public Boolean
AccessCardEnabled	Boolean	Public Boolean

Delphi

```
Var
Arac: THGSArac;
iErr, InxNm: Integer;
CardID: String;
Devam : Boolean;
Arac.Name := ….;
Arac.TimeAccessMask[0] := ….;
Arac.TimeAccessMask[1] := ….;
Arac.TimeAccessMask[2] := ….;
Arac.TimeAccessMask[3] := ….;
Arac.TimeAccessMask[4] := ….;
Arac.TimeAccessMask[5] := ….;
Arac.TimeAccessMask[6] := ….;
Arac.Daire := ….;
Arac.AracNo := ….;
Arac.EndDate := ….;
Arac.AccessDevice := ….;
Arac.APBEnabled := false ;//ya da true;
Arac.ATCEnabled := false; // ya da true;
Rdr.AddHGSWhitelist(Arac,InxNm);
```

C#

```
int iErr;
int InxNm;
String CardID ;
Boolean Devam;
THGSArac Arac;
Arac = new THGSArac();
Arac.CardID = CardID;
Arac.Name = ….;
Arac.TimeAccessMask[0] = (byte) ….;
Arac.TimeAccessMask[1] = (byte) ….;
Arac.TimeAccessMask[2] = (byte) ….;
Arac.TimeAccessMask[3] = (byte) ….;
Arac.TimeAccessMask[4] = (byte) ….;
Arac.TimeAccessMask[5] = (byte) ….;
Arac.TimeAccessMask[6] = (byte) ….;
Arac.Daire = (ushort) ….;
Arac.AracNo = (byte) ….;
Arac.EndDate = Convert.ToDateTime(….);
Arac.AccessDevice = true; // ya da false;
Arac.APBEnabled = true; // ya da false
Arac.ATCEnabled = true; // ya da false;
```

HGS’de TAG Düzeltme İşlemi.

HGS sistemine TAG düzeltme işlemi “***EditHGSWhitelistWithCardID***” isimli fonksiyondur. Bu fonksiyon “***THGSArac***” ve “***Integer***” tipinde iki parametre ile çalışır. Fonksiyonun genel kullanımı *Rdr.EditHGSWhitelistWithCardID(Arac,InxNm); “**THGSArac”*** sınıf dizisinin parametrik yapıları yukarıda belirtilmiştir. Bu fonksiyon sadece uygulama tipi olarak " ***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Arac: THGSArac;
InxNm: Integer;
CardID: String;
Arac.CardID := CardID;
Arac.Name := ….;
Arac.TimeAccessMask[0] := ….;
Arac.TimeAccessMask[1] := ….;
Arac.TimeAccessMask[2] := ….;
Arac.TimeAccessMask[3] := ….;
Arac.TimeAccessMask[4] := ….;
Arac.TimeAccessMask[5] := ….;
Arac.TimeAccessMask[6] := ….;
Arac.Daire := ….;
Arac.AracNo := ….
Arac.EndDate := ….
Arac.AccessDevice := ….;
Arac.APBEnabled := ….;
Arac.ATCEnabled := ….;
Rdr.EditHGSWhitelistWithCardID(Arac, InxNm);
```

C#

```
THGSArac Arac;
Arac = new THGSArac();
int iErr, i, cnt, InxNm;
String CardID ;
Arac.CardID := CardID;
Arac.Name := …;
Arac.TimeAccessMask[0] := …;
Arac.TimeAccessMask[1] := …;
Arac.TimeAccessMask[2] := …;
Arac.TimeAccessMask[3] := …;
Arac.TimeAccessMask[4] := …;
Arac.TimeAccessMask[5] := …;
Arac.TimeAccessMask[6] := …;
Arac.Daire := …;
Arac.AracNo := …;
Arac.EndDate := …;
Arac.AccessDevice := …;
Arac.APBEnabled := …;
Arac.ATCEnabled := …;
Rdr.EditHGSWhitelistWithCardID(Arac, InxNm);
```

HGS’de TAG Silme İşlemi.

HGS sistemine TAG silme işlemi “***DeleteHGSWhitelistWithCardID***” isimli fonksiyondur. Bu fonksiyon “***Integer***” tipinde iki parametre ile çalışır. Fonksiyonun genel kullanımı *Rdr. DeleteHGSWhitelistWithCardID(CardID, InxNm);* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

Var

CardID: String;

InxNm: Integer;

Rdr.DeleteHGSWhitelistWithCardID(CardID, InxNm);

C#

```
String CardID;
int InxNm;
rdr.DeleteHGSWhitelistWithCardID(CardID, out InxNm);
```

HGS’de TAG Bulma İşlemi.

HGS sistemine TAG bulma işlemi “***GetHGSWhitelistWithCardID***” isimli fonksiyondur. Bu fonksiyon 1 “***THGSArac***” ve 2 “***Integer***” tipinde iki parametre ile çalışır. Fonksiyonun genel kullanımı *Rdr.GetHGSWhitelistWithCardID(CardID, Arac, InxNm);* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır. “***THGSArac”*** sınıf dizi yapıları yukarıda belirtilmiştir. Fonksiyondan gelen araç bilgileri “***THGSArac”*** sınıf dizisi elemanlarına atanır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
Arac: THGSArac;
InxNm: Integer;
CardID: String;
Rdr.GetHGSWhitelistWithCardID(CardID, Arac, InxNm);
…… := Trim(Arac.Name);
…… := Arac.TimeAccessMask[0];
…… := Arac.TimeAccessMask[1];
…… := Arac.TimeAccessMask[2];
…… := Arac.TimeAccessMask[3];
…… := Arac.TimeAccessMask[4];
…… := Arac.TimeAccessMask[5];
…… := Arac.TimeAccessMask[6];
…… := Arac.Daire;
…… := Arac.AracNo;
…… := Arac.EndDate;
…… := Arac.AccessDevice;
…… := Arac.APBEnabled;
…… := Arac.ATCEnabled;
…… := IntToStr(InxNm);
```

C#

```
THGSArac Arac;
int cnt, InxNm;
String CardID;
rdr.GetHGSWhitelistWithDaireArac((ushort)….,(byte)…., out Arac, out InxNm);
….. = Arac.CardID;
….. = Arac.Name.Trim();
….. = Arac.TimeAccessMask[0];
…..= Arac.TimeAccessMask[1];
….. = Arac.TimeAccessMask[2];
…..= Arac.TimeAccessMask[3];
….. = Arac.TimeAccessMask[4];
….. = Arac.TimeAccessMask[5];
….. = Arac.TimeAccessMask[6];
….. = Arac.Daire;
….. = Arac.AracNo;
….. = Arac.EndDate;
….. = Arac.AccessDevice;
….. = Arac.APBEnabled;
….. = Arac.ATCEnabled;
….. = Convert.ToString(InxNm);
```

HGS’de Daire Araç Silme İşlemi.

HGS sistemine daire araç silme işlemi “***DeleteHGSWhitelistWithDaireArac***” isimli fonksiyondur. Bu fonksiyon 3 paramete ile çalışır. Fonksiyonun genel kullanımı *Rdr.GetHGSWhitelistWithCardID(IntegerDeğer, THGSAracDeğer, IntegerDeğer);* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.DeleteHGSWhitelistWithDaireArac(edtHGSDaireNo.Value,edtHGSAracNo.Value,InxNm)
```

C#

```
rdr.DeleteHGSWhitelistWithDaireArac((ushort)….., (byte)….., out InxNm)
```

HGS’de Daire Araç Bulma İşlemi.

HGS sistemine daire araç bulma işlemi “***GetHGSWhitelistWithDaireArac***” isimli fonksiyondur. Bu fonksiyon 4 paramete ile çalışır. Fonksiyonun genel kullanımı *Rdr.GetHGSWhitelistWithDaireArac(IntegerDeğer,Integerdeğer, THGSAracDeğer, IntegerDeğer)* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
Arac: THGSArac;
InxNm: Integer;
CardID: String;
Rdr.GetHGSWhitelistWithDaireArac(edtHGSDaireNo.Value,edtHGSAracNo.Value, Arac, InxNm);
….. := Arac.CardID;
….. := Trim(Arac.Name);
….. := Arac.TimeAccessMask[0];
….. := Arac.TimeAccessMask[1];
….. := Arac.TimeAccessMask[2];
….. := Arac.TimeAccessMask[3];
….. := Arac.TimeAccessMask[4];
….. := Arac.TimeAccessMask[5];
….. := Arac.TimeAccessMask[6];
….. := Arac.Daire;
….. := Arac.AracNo;
….. := Arac.EndDate;
….. := Arac.AccessDevice;
….. := Arac.APBEnabled;
….. := Arac.ATCEnabled;
….. := IntToStr(InxNm);
```

C#

```
THGSArac Arac;
int iErr, i, cnt, InxNm;
String CardID;
rdr.GetHGSWhitelistWithDaireArac((ushort)IntegerDeğer,(byte)IntegerDeğer, out Arac, out InxNm);
…..= Arac.CardID;
…..= Arac.Name.Trim();
….. = Arac.TimeAccessMask[0];
…..= Arac.TimeAccessMask[1];
….. = Arac.TimeAccessMask[2];
….. = Arac.TimeAccessMask[3];
…..= Arac.TimeAccessMask[4];
….. = Arac.TimeAccessMask[5];
…..= Arac.TimeAccessMask[6];
….. = Arac.Daire;
…..= Arac.AracNo;
….. = Arac.EndDate;
Arac.AccessDevice;
Arac.APBEnabled ;
Arac.ATCEnabled ;
….. = Convert.ToString(InxNm);
```

HGS’de Cihazda Tanımlı Kart Sayısını Bulma.

HGS sistemine TAG ekleme işlemi “***GetWhitelistCardIDCount***” isimli fonksiyondur. Bu fonksiyon parametresiz çalışır. Fonksiyonun genel kullanımı *GetWhitelistCardIDCount() gibidir.* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
…… := GetWhitelistCardIDCount();
```

C#

```
…… = GetWhitelistCardIDCount();
```

HGS’de Daire Araç Otopark Hakkı Alma.

HGS sistemine daire araç otopark hakkı alma işlemi “***GetHGSDaireParkHak***” isimli fonksiyondur. Bu fonksiyon iki parametre ile çalışır. Fonksiyonun genel kullanımı *Rdr.GetHGSDaireParkHak(IntegerDeğer,Bytedeğer) şeklindedir.* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır. Fonksiyon ile gönderilen daire numarasını temsil eden Integer değere, karşılık gelen daire hakkı, byte değere atanır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Var
Hak:Byte;
Rdr.GetHGSDaireParkHak(edtHGSDaireHK.Value,Hak);
….:=hak;
```

C#

```
byte hak=0;
rdr.GetHGSDaireParkHak((uint)txtDaireAracHakkiOtoparkHakkiDaire.Value,out hak)
…..:=hak;
```

HGS’de Daire Araç Otopark Hakkı Gönderme.

HGS sistemine daire araç otopark hakkı alma işlemi “***SetHGSDaireParkHak***” isimli fonksiyondur. Bu fonksiyon iki parametre ile çalışır. Fonksiyonun genel kullanımı *Rdr. SetHGSDaireParkHak(IntegerDeğer,Bytedeğer) şeklindedir.* Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır. Fonksiyon ile gönderilen daire numarasını maksimum araç hakkı tanır.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
Rdr.SetHGSDaireParkHak(IntegerDeğer,ByteDeğer);
```

C#

```
rdr.SetHGSDaireParkHak((byte)IntegerDeğer, (uint)byteDeğer);
```

HGS’de Giriş Çıkış Bilgisi Alma.

HGS sistemine giriş çıkış bilgileri alma fonksiyonu “***ReadHGSRecords***” Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır. Fonksiyon ile gönderilen daire numarasını maksimum araç hakkı tanır. Fonksiyonun genel kullanımı *Rdr.ReadHGSRecords(CardinalDeğer, CardinalDeğer, THGSRecordsDeğer);* şeklindedir. Cihazdan gelen bilgiler “***THGSRecords”*** tipindeki dizi değişkenin elemanlarına atanır. “***THGSRecords”*** parametrik yapıları aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

THGSRecords Parametrik Yapıları		
	Delphi	C#
Count	Cardinal	public uint
raDeviceRecs	array of TOneHGSRecord	public TOneHGSRecord[]

TOneHGSRecord Parametrik Yapıları		
	Delphi	C#
CardID	string	public string
DoorNo	Byte	public byte
RecordType	Byte	public byte
TimeDate	TDateTime	public DateTime

Delphi

```
var
tempRecs: THGSRecords;
i: Integer;
strCardId: String;
Rdr.ReadHGSRecords(CardinalDeğer, CardinalDeğer, tempRecs);
for i := 0 to tempRecs.Count - 1 do
begin
….. := tempRecs.raDeviceRecs[i].CardID;
….. := tempRecs.raDeviceRecs[i].DoorNo;
….. := tempRecs.raDeviceRecs[i].RecordType) ;
….. := tempRecs.raDeviceRecs[i].TimeDate);
end;
```

C#

```
THGSRecords tempRecs;
String strCardId;
rdr.ReadHGSRecords((uint)IntegerDeğer, (uint)IntegerDeğer , out tempRecs) ;
for (int i = 0; i <= tempRecs.Count; i++)
{
…… = tempRecs.raDeviceRecs[i].CardID;
…… =strCardId ;
…… =Convert.ToString(tempRecs.raDeviceRecs[i].DoorNo) ;
…… = Convert.ToString(tempRecs.raDeviceRecs[i].RecordType) ;
…… = Convert.ToString(tempRecs.raDeviceRecs[i].TimeDate) ;
}
```

HGS’de Giriş Çıkış Bilgisi Gönderme.

HGS sistemine giriş çıkış bilgileri transfer etme fonksiyonu “***TransferHGSRecords***” Bu fonksiyon sadece uygulama tipi olarak "***HGS***" tipinde çalışır. Fonksiyon ile gönderilen daire numarasını maksimum araç hakkı tanır. Fonksiyonun genel kullanımı *Rdr.TransferHGSRecords(tempRecs);* şeklindedir. Cihaza gönderilecek bilgiler “***THGSRecords”*** tipindeki dizi değişkenin elemanlarına değerler atanarak gönderilir. “***THGSRecords”*** parametrik yapıları aşağıda belirtilmiştir.

(! Komut çalıştırılmadan önce bağlantı olup olmadığı kontrol edilmelidir.)

Delphi

```
var
tempRecs: THGSRecords;
i: Integer;
strCardId: String;
Rdr.TransferHGSRecords(tempRecs);
for i := 0 to tempRecs.Count - 1 do
begin
strCardId := tempRecs.raDeviceRecs[i].CardID;
tempRecs.raDeviceRecs[i].DoorNo := …..;
tempRecs.raDeviceRecs[i].RecordType :=……;
tempRecs.raDeviceRecs[i].TimeDate) :=…..;
end;
```

C#

```
THGSRecords tempRecs ;

rdr.TransferHGSRecords(out tempRecs);
for (int i = 0; i <= tempRecs.Count ; i++)
{
tempRecs.raDeviceRecs[i].DoorNo)  = ……;
tempRecs.raDeviceRecs[i].RecordType) = ……;
tempRecs.raDeviceRecs[i].TimeDate) = ……;
}
```

Component Events (Olaylar)

- OnRxCardID: "***RxCarID***" isimli olayı tetikleyen nesnedir. Bu event karttan gelen "***CardID***" sini argüman olarak döndürür.

- OnRxTurnStileTurn: "***RxTurnStileTurn***" isimli olayı tetikleyen nesnedir. Bu event turnikenin döndüğü durumlarda gerçekleşen olaydır. Event "***CardID***" ve "***Success***" isimli iki parametreyi argüman olarak döndürür. "***Success"*** boolean türünde , ***"CardID"*** ise string türündedir. Bu olay "uygulama ayalarında" yer alan "Geçiş tipleri" seçeneğinde bulunan "Turnike" seçeneği ile çalışır.

- OnSerialReadStr: ***"RxSerialReadStr"*** isimli olayı tetikleyen nesnedir. Bu event seri porttan gelen verileri okumak için kullanılmaktadır. Event "***byte***" türünde ***"SerialPortNo"*** isminde ve "***string***" tipinde ***"strData"*** ismindeki değerleri argüman olarak döndürür. ***"SerialPortNo"*** okunan bilginin hangi porttan okunduğunun temsil eder. ***"StrData"*** okunan verinin string türünden değerini temsil eder.

- OnRxDoorOpenAlarm: "***RxDoorOpenAlarm***" isimli olayı tetikleyen nesnedir. Bu event belirtilen süres içerisinde kapının kapanmadığı durumlarda tetiklenir. Kapının belirtilen süre zarfında kapanmaması durumunda bu olay devreye girer. Event "***boolean***" tipinde "***DoorOpen***" ve "***integer***" tipinde "***OpenTime***" isimli değerleri argüman olarak geri döndürür. "***OpenTime***" değeri; kapının açık kaldığı zaman bilgisini milisaniye cinsinden tutar. Bu olay "uygulama ayalarında" yer alan "Geçiş tipleri" seçeneğinde bulunan "Kapı Alarmı" seçeneği ile çalışır.

- OnTagRead: "***RxTagRead***" isimli olayı tetikleyen nesnedir. Bu event "***HGS***" sistemleri için kullanılmaktadır. "***byte***" tipinde "***IO***" ve "***string***" tipinde "***TagId***" değerlerini argüman olarak döndürür. "***IO***" giriş veya çıkış okuyucularını temsil eder. 0 giriş 1 çıkış okuyucularını temsil eder. "***TagId***" ise okunan etiketin (TAG) ID sini temsil eder.

- OnPassWordRead: "***RxOnPasswordRead***" isimli olayı tetikleyen nesnedir. Bu event cihaz panelinden girilen şifreyi almak için kullanılmaktadır. "***byte***" tipinde "***PassType***" isminde, "***cardinal***" tipinde "***Password***" isminde ve "***cardinal***" tipinde "***Code***" ismindeki değerleri argüman olarak döndürür. "Code" değeri en fazla 65535 alabilmektedir.

- OnCardRead: "***RxOnCardRead***" isimli olayı tetikleyen nesnedir. Bu event cihazdan okutulan kartların kart Id lerini almak için kullanılan eventdır. Bu event CardID isimli parametrenin içinden alınarak kullanılır.