	مستند ارتباط با پوز DLL	
	نسخه: ۱,۰,۰,۳	صفحه: ۱

۱. آشنایی با PosInterface.dll


PosInterface.dll یک Net Dll است که امکانات موردنظر برای ارتباط یک برنامه با POS را فراهم می‌کند. برنامه‌هایی که با تکنولوژی Net. تهیه می‌شوند بعنوان مثال برنامه‌هایی که به زبان‌های C#.Net یا VB.Net هستند، می‌توانند مستقیماً از این Dll استفاده کنند اما برنامه‌هایی که مبتنی بر تکنولوژی Net. نیستند، می‌بایست از شیء COM تهیه شده از این Dll استفاده کنند. در ادامه نحوه نصب این Dll بصورت COM در ویندوز توضیح داده می‌شود.

۲. نصب COM

برای نصب COM برای یک Net Dll. ابتدا می‌بایست Net Framework را بر روی سیستم‌تان نصب نمایید. نسخه Net Framework. می‌بایست با نسخه‌ای از Net Framework. که Dll بوسیله آن Build شده است یکسان باشد. فرض کنید شما می‌خواهید PosInterface.dll ای که با Net Framework 2 تهیه شده است را نصب کنید. در اینصورت می‌بایست ابتدا اگر Net Framework 2. بر روی سیستم‌تان نصب نیست، آن را نصب کنید و سپس فرمان زیر را از طریق محیط Command در ویندوز اجرا کنید:

[PathToRegAsm]\RegAsm.exe [PathToDll]\PosInterface.dll /codebase /tlb

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
			تاریخ تهیه: ۱۳۹۲/۰۹/۲۴
			تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲

	مستند ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۲

که در [PathToRegAsm] و [PathToDll] به ترتیب آدرس شاخه‌های برنامه RegAsm.exe و PosInterface.dll هستند. بعنوان مثال اگر شاخه محتوی PosInterface.dll به آدرس C:\PosInterface باشد و Net Framework در شاخه پیش فرض نصب شده باشد، فرمان موردنظر به شکل زیر خواهد بود:


"C:\Windows\Microsoft.NET\Framework\v2.0.50727\RegAsm.exe" "C:\PosInterface\PosInterface.dll" /codebase /tlb

۳. اطمینان از صحت نصب COM

برای اطمینان از صحت نصب COM می‌توانید موارد زیر را بررسی کنید:

- می‌بایست پس از اجرای فرمان در محیط Command پیغام رجیستر شدن موفقیت‌آمیز کلاس‌ها نمایش داده شود.
- می‌بایست پس از اجرای فرمان، در آدرسی که Dll قرار دارد فایل PosInterface.tlb ساخته شود. (بهتر است قبل از اجرای فرمان، اگر این فایل وجود دارد آن را پاک کنید تا از صحت فرآیند نصب اطمینان یابید).
- می‌بایست پس از اجرای فرمان، با ابزار RegDllView.exe و تیکه Dll های نصب شده در ویندوز را مشاهده می‌کنید PosInterface.dll نیز نمایش داده شود. لازم است که اگر قبلاً نسخه‌های قبلی این Dll را بر روی سیستم‌تان نصب کرده‌اید، قبل از نصب نسخه جدید، نسخه قبلی را بوسیله همین ابزار پاک

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
			تاریخ تهیه: ۱۳۹۲/۰۹/۲۴
			تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲

	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۳

کنید. (RegDllView.exe یک ابزار رایگان برای مشاهده و حذف DLL های رجیستر شده در ویندوز است)

۴. استفاده از COM

بعد از رجیستر شدن موفقیت آمیز COM نوبت به استفاده از آن در زبان برنامه نویسی مورد نظر است. محیط های برنامه نویسی مختلف، به روش های مختلف امکان استفاده از COM در برنامه ها را می دهند و برای اینکار ویزاردهایی دارند. بعنوان مثال در محیط Delphi می بایست از منوی Project>References... استفاده نمایید و یا در محیط VB 6.0 می بایست از منوی Project>Import Type Library... استفاده نمایید و از لیست DLL های رجیستر شده PosInterface را انتخاب نمایید.


۵. نحوه کار با کلاس های PosInterface

برای استفاده از کلاس های PosInterface می توانید از راهنمای زیر استفاده نمایید:

۱. یک شیء از نوع PCPos بسازید:

```
PCPos pcPos = new PCPos();
```

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
		تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲	تاریخ تهیه: ۱۳۹۲/۰۹/۲۴

	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۴

۲. چنانچه قصد دارید از طریق شبکه به پوز متصل شوید، تابع InitLAN را صدا بزنید و ip دستگاه POS را بعنوان ورودی وارد نمایید. برای پارامتر پورت همواره مقدار ۱۷۰۰۰ را وارد نمایید. بعنوان مثال اگر ip پوز برابر با ۱۹۸,۱۰,۱۶۸,۱۹۲ باشد دستور موردنظر، عبارتست از:


```
pcPos.InitLAN("192.168.10.98", 17000);
```

اما اگر قصد دارید برای اتصال به POS از پورت سریال استفاده نمایید. تابع InitSerial را صدا بزنید و شماره COM ای که پورت سریال را به آن وصل کرده‌اید را بعنوان ورودی تابع وارد نمایید و برای پارامتر BaudRate همواره مقدار ۱۱۵۲۰۰ را وارد نمایید. بعنوان مثال اگر شماره پورت سریال COM2 باشد از دستور زیر استفاده نمایید:

```
pcPos.InitSerial("COM2", 17000);
```

۳. میزان حداکثر زمانی که برنامه پس از ارسال درخواست برای POS می‌بایست برای دریافت پاسخ هر تراکنش آن معطل بماند را مشخص نمایید. مقدار پیش فرض ۶۰۰۰۰ میلی ثانیه (۱ دقیقه) است. شما می‌توانید هر مقداری بین ۶۰۰۰۰ تا ۳۰۰۰۰۰ میلی ثانیه (۱ الی ۵ دقیقه) را انتخاب کنید. دقت نمایید چنانچه مقداری خارج از این بازه را وارد نمایید، مقدار جدید اعمال نخواهد شد. همچنین دقت نمایید که این مقدار حداکثر زمانی است که برنامه برای دریافت پاسخ تراکنش منتظر می‌ماند و اگر POS زودتر پاسخ را ارسال کند، همان موقع پاسخ را در برنامه دریافت خواهید کرد. بعنوان مثال چنانچه بخواهید برای هر تراکنش Timeout برابر با ۲ دقیقه انتخاب کنید می‌بایست از دستور زیر استفاده کنید:

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
			تاریخ تهیه: ۱۳۹۲/۰۹/۲۴
		تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲	

	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۵

pcPos.TimeoutPerTransaction = 120000;

۴. چنانچه قصد دارید یک تراکنش خرید انجام دهید و می‌خواهید اینکار را بصورت سنکرون انجام دهید، یعنی می‌خواهید فراخواننده تا رسیدن پاسخ بلاک شود تابع DoSyncPayment را فراخوانی نمایید. بعنوان مثال اگر می‌خواهید خریدی با مبلغ ۱۰۰۰۰ ریال و با شماره فاکتور ۱۲۳۲ انجام دهید و مبلغ مورد نظر در دو شبای مختلف تسهیم شود، می‌بایست از دستور زیر استفاده نمایید:

```
PaymentResult result = pcPos.DoSyncPayment("10000",
"IR530140040000310xxxxx4961;7000;IR92016000000000xxxxx6077;3000", "1232", DateTime.Now);
```


❖ در صورتی که تسهیم وجوه وجود ندارد می‌توانید String خالی در این آرگومان قرار دهید.

❖ در استفاده از تسهیم وجوه باید مجموع مبالغ تسهیم با مبلغ کل برابر باشد، در غیر اینصورت پیغام خطا صادر می‌شود.

همانطور که مشاهده می‌فرمایید خروجی تراکنش در قالب یک شیء PaymentResult برمی‌گردد. از طریق Property های این شیء می‌توانید به جزییات نتیجه فراخوانی درخواست دسترسی پیدا کنید. این Property ها عبارتند از:

نام	توضیحات
ErrorCode	کد خطا، در صورتی که تراکنش موفقیت‌آمیز باشد برابر با ۰ خواهد بود.
ErrorMsg	در صورتیکه تراکنش موفقیت‌آمیز نباشد برابر با توضیح خطا است.

نام تهیه کننده	نوع سند	واحد تهیه کننده	وضعیت
بزرگ زاده		تحقیق و توسعه نرم افزار	فعال
تاریخ تهیه: ۱۳۹۲/۰۹/۲۴	تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲		


	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۶

InvoiceNumber همان شماره فاکتوری است که در فراخوانی تابع استفاده کرده‌اید.	
MerchantId	کد پذیرنده (فروشگاه)
TerminalId کد ترمینال (POS)	
RRN شماره مرجع تراکنش	
Stan شماره پیگیری	
CardNumber شماره کارت (بنابر قوانین، نمی‌بایست شماره کارت بصورت کامل ارایه شود و در این فیلد برخی کاراکترهای میانی شماره کارت پر نمی‌شود)	
DateTime تاریخ انجام تراکنش	
PaymentAmount مبلغ تراکنش	
MessageId شماره درخواست	

۵. چنانچه قصد دارید یک تراکنش خرید انجام دهید و می‌خواهید اینکار را بصورت آسنکرون انجام دهید، یعنی می‌خواهید فراخواننده بلاک نشود تابع DoAsyncPayment را فراخوانی نمایید. بعنوان مثال اگر می‌خواهید خریدی با مبلغ ۱۰۰۰۰ ریال و با شماره فاکتور ۱۲۳۲ انجام دهید، و می‌خواهید نتیجه عملیات را در کنسول چاپ کنید می‌توانید از کد زیر استفاده نمایید:

```
public class MyHandler : ITransactionDoneHandler
{
```

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
			تاریخ تهیه: ۱۳۹۲/۰۹/۲۴
			تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲

	مستند DLL ارتباط با پوز	
	صفحه: ۷	نسخه: ۱,۰,۰,۳

```

public void OnTransactionDone(TransactionResult result)
{
    Console.WriteLine(result.ToString());
}

public void OnFinish(string message)
{
    Console.WriteLine(message);
}
}

```

...

```

MyHandler handler = new MyHandler();
pcPos.DoAsyncPayment("10000", "IR530140040000310xxxxx4961;7000;IR9201600000000000xxxxx6077;3000", "1232", DateTime.Now, handler);


```

❖ در صورتی که تسهیم وجوه وجود ندارد می توانید String خالی در این آرگومان قرار دهید.

❖ در استفاده از تسهیم وجوه باید مجموع مبالغ تسهیم با مبلغ کل برابر باشد، در غیر اینصورت پیغام خطا صادر می شود.

❖ در صورتی که به هر دلیل پاسخی از POS دریافت نشد می توانید پس از اتمام زمان انتظار از دستور استعلام وضعیت آخرین خرید استفاده کنید. سناریوی استفاده از این دستور به عهده کاربر dll می باشد و مثلاً می تواند به هر تعدادی که لازم می داند از این دستور استفاده کند تا نتیجه مطلوب سناریوی خود را به دست آورد. بدیهی است در صورتی که دستور خرید دیگری به پوز ارسال شود، وضعیت تراکنش خرید قبلی با وضعیت تراکنش جدید جایگزین خواهد شد و

نام تهیه کننده	نوع سند	واحد تهیه کننده	وضعیت
بزرگ زاده		تحقیق و توسعه نرم افزار	فعال
تاریخ تهیه: ۱۳۹۲/۰۹/۲۴	تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲		


	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۸

چنانچه نتیجه تراکنش قبلی همچنان نامشخص باشد، نتیجه آن جزء مغایرت ها خواهد بود که در صورت نیاز باید از روشهای موجود مغایرت گیری استفاده شود.

همانطور که مشاهده می فرمایید نخست یک کلاس با نام MyHandler تعریف شده است که اینترفیس ITransactionDoneHandler را پیاده سازی کرده است. این اینترفیس دو متد با نام های OnTransactionDone و OnFinish دارد. OnTransactionDone وقتی فراخوانی می شود که پاسخ یکی از تراکنش های درخواستی آماده شده باشد و OnFinish پس از پایان یافتن کل عملیات فراخوانی می شود. دقت نمایید که در حالت کلی یک درخواست ممکن است شامل فراخوانی چندین تراکنش باشد. بعنوان مثال در مورد پرداخت قبض این امکان وجود دارد که با یک فراخوانی تابع، برای پرداخت چندین قبض درخواست نمایید. البته در مورد خرید این امکان وجود ندارد و همواره با هربار فراخوانی تابع تنها یک تراکنش خرید انجام خواهد شد.

در فراخوانی تابع DoASyncPayment می بایست یک شیء از نوع ITransactionDoneHandler نیز بعنوان ورودی ارسال کنید. در واقع توابع OnTransactionDone و OnFinish آن شیء بعدا و بصورت آسنکرون فراخوانی خواهند شد. چنانچه تراکنش خرید موفقیت آمیز باشد یکبار تابع OnTransactionDone و یکبار تابع OnFinish فراخوانی می شود. در مورد تراکنش های غیرمجاز فقط تابع OnFinish فراخوانی می شود و متن پیغام خطا در ورودی این تابع فراهم می شود.

نام تهیه کننده	نوع سند	واحد تهیه کننده	وضعیت
بزرگ زاده		تحقیق و توسعه نرم افزار	فعال
تاریخ تهیه: ۱۳۹۲/۰۹/۲۴	تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲		

	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۹


نتیجه انجام تراکنش در ورودی تابع OnTransactionDone که از نوع TransactionResult است فراهم می‌شود. TransactionResult نوع والد پاسخ‌های تراکنش‌ها است و انواع PaymentResult و BillPaymentResult از آن ارث می‌برند.

همانطور که در مثال بالا می‌بینید یکی از متدهای TransactionResult تابع ToString است که مقادیر کلیه فیلدهای موجود در خروجی را بصورت رشته‌ای ارائه می‌کند و می‌توانید برای لاگ زدن یا چاپ خروجی از آن استفاده نمایید. البته در مثال بالا، این امکان نیز وجود داشت که result را به PaymentResult، Cast کنید و به هر کدام از مقادیر فیلدها مستقیماً دسترسی پیدا کنید.

۶. چنانچه قصد دارید یک تراکنش پرداخت قبض انجام دهید، می‌توانید اینکار را بصورت آسنکرون با استفاده از تابع DoAsyncBillPayment انجام دهید. این امکان وجود دارد که درخواست پرداخت چندین قبض را با هم صادر نمایید. بعنوان مثال اگر بخواهید دو قبض که شناسه پرداخت و شناسه قبض یکی به ترتیب برابر با ۱۲۰۲۵۸ و ۲۵۳۸۰۵۷۴۰۹۱۵۸ است و شناسه پرداخت و شناسه قبض دیگری، به ترتیب برابر با ۱۲۰۳۳۳ و ۲۴۰۲۶۰۰۹۰۹۱۵۹ است را پرداخت نمایید می‌توانید از فراخوانی زیر استفاده نمایید:

```
MyHandler handler = new MyHandler();
pcPos.DoAsyncBillPayment("120258;120333",
    "2538057409158;2402600909159",
    DateTime.Now, handler);
```

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
			تاریخ تهیه: ۱۳۹۲/۰۹/۲۴
			تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲


	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۱۰

همانطور که در توضیحات بند قبل ذکر شد، می‌بایست یک شیء از نوع `ITransactionDoneHandler` نیز بعنوان ورودی ارسال کنید که توابع `OnTransactionDone` و `OnFinish` آن شیء بعداً و بصورت آسنکرون فراخوانی خواهند شد. نتیجه هرکدام از تراکنش‌های پرداخت قبض با یکبار فراخوانی `OnTransactionDone` فراهم می‌شود و در انتهای کار نیز تابع `OnFinish` فراخوانی می‌شود. چنانچه مشکلی پیش بیاید که انجام مابقی تراکنش‌ها ممکن نباشد ممکن است همان ابتدای کار و یا پس از انجام چند تراکنش اول، ادامه کار ملغی شده و `OnFinish` ارسال شود.

وقتی شما تابع `DoASyncBillPayment` را فراخوانی می‌نمایید اشیاء ورودی تابع `OnTransactionDone` در واقع از نوع `BillPaymentResult` هستند و می‌توانید آنها را به این نوع، `Cast` کنید. از طریق `Property` های این شیء می‌توانید به جزییات نتیجه فراخوانی درخواست دسترسی پیدا کنید. این `Property` ها عبارتند از:

نام	توضیحات
<code>ErrorCode</code>	کد خطا، در صورتی که تراکنش موفقیت‌آمیز باشد برابر با ۰ خواهد بود.
<code>ErrorMsg</code>	در صورتیکه تراکنش موفقیت‌آمیز نباشد برابر با توضیح خطا است.
<code>BillId</code>	همان شماره قبضی است که در فراخوانی تابع استفاده کرده‌اید.

نام تهیه کننده	نوع سند	واحد تهیه کننده	وضعیت
بزرگ زاده		تحقیق و توسعه نرم افزار	فعال
تاریخ تهیه: ۱۳۹۲/۰۹/۲۴	تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲		

	مستند DLL ارتباط با پوز	
	نسخه: ۱,۰,۰,۳	صفحه: ۱۱

PaymentId همان شماره پرداختی است که در فراخوانی تابع استفاده کرده‌اید.	
MerchantId کد پذیرنده (فروشگاه)	TerminalId کد ترمینال (POS)
RRN شماره مرجع تراکنش	
Stan شماره پیگیری	
CardNumber شماره کارت (بنابر قوانین، نمی‌بایست شماره کارت بصورت کامل ارایه شود و در این فیلد برخی کاراکترهای میانی شماره کارت پر نمی‌شود)	
DateTime تاریخ انجام تراکنش	
PaymentAmount مبلغ تراکنش	
MessageId شماره درخواست	

نکته: در هر بار فراخوانی تابع، می‌توانید درخواست پرداخت حداکثر ۱۰ قبض را صادر نمایید.

وضعیت	واحد تهیه کننده	نوع سند	نام تهیه کننده
فعال	تحقیق و توسعه نرم افزار		بزرگ زاده
			تاریخ تهیه: ۱۳۹۲/۰۹/۲۴
		تاریخ ویرایش: ۱۳۹۴/۱۰/۰۲	