# PAY AT TABLE API

## Contents

# Overview

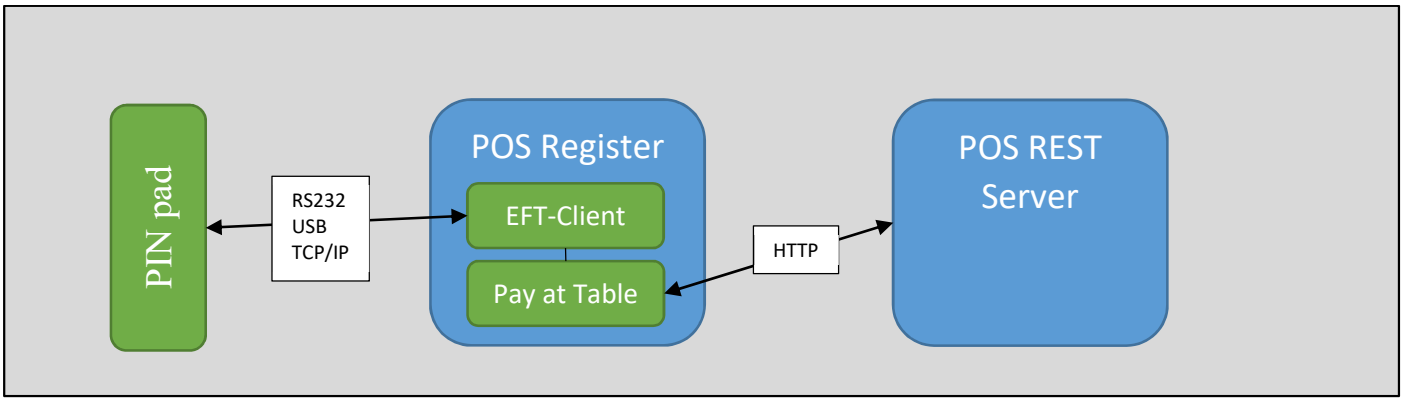
The Pay at Table API provides a common interface for the PIN pad to utilise the EFT-Client to retrieve available tables and orders so payment functions (e.g. tender, customer receipt etc.) can be performed by an operator on the PIN pad without using the main POS UI.

The Pay at Table client requires the POS to act a data source so that it can retrieve information about available tables, orders, payment options etc.

The Pay at Table client supports two data source options for the POS; a REST server or directly through the existing PC-EFTPOS interface.

## REST Server

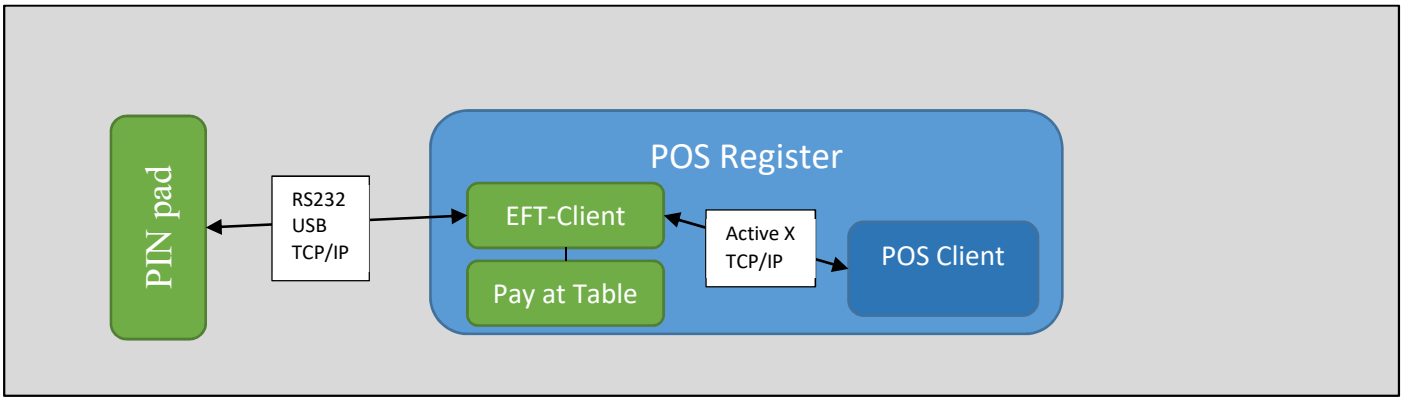
When in REST server mode the Pay at Table extension will connect directly to the POS REST Server.

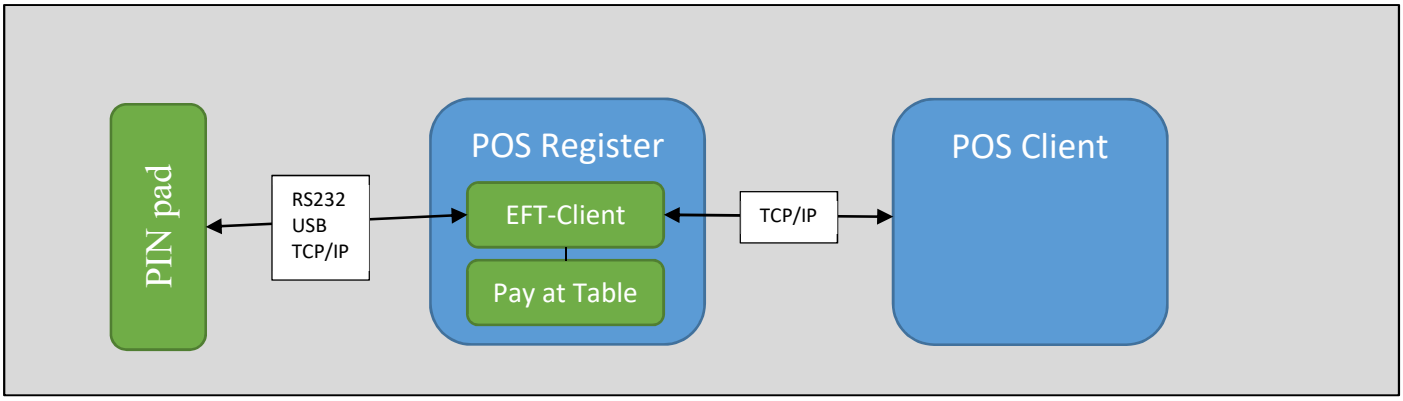



## PC-EFTPOS Interface

When in POS mode the Pay at Table extension will utilise the existing interface between the POS and EFT-Client (i.e. the interface used to perform a transaction using PC-EFTPOS).

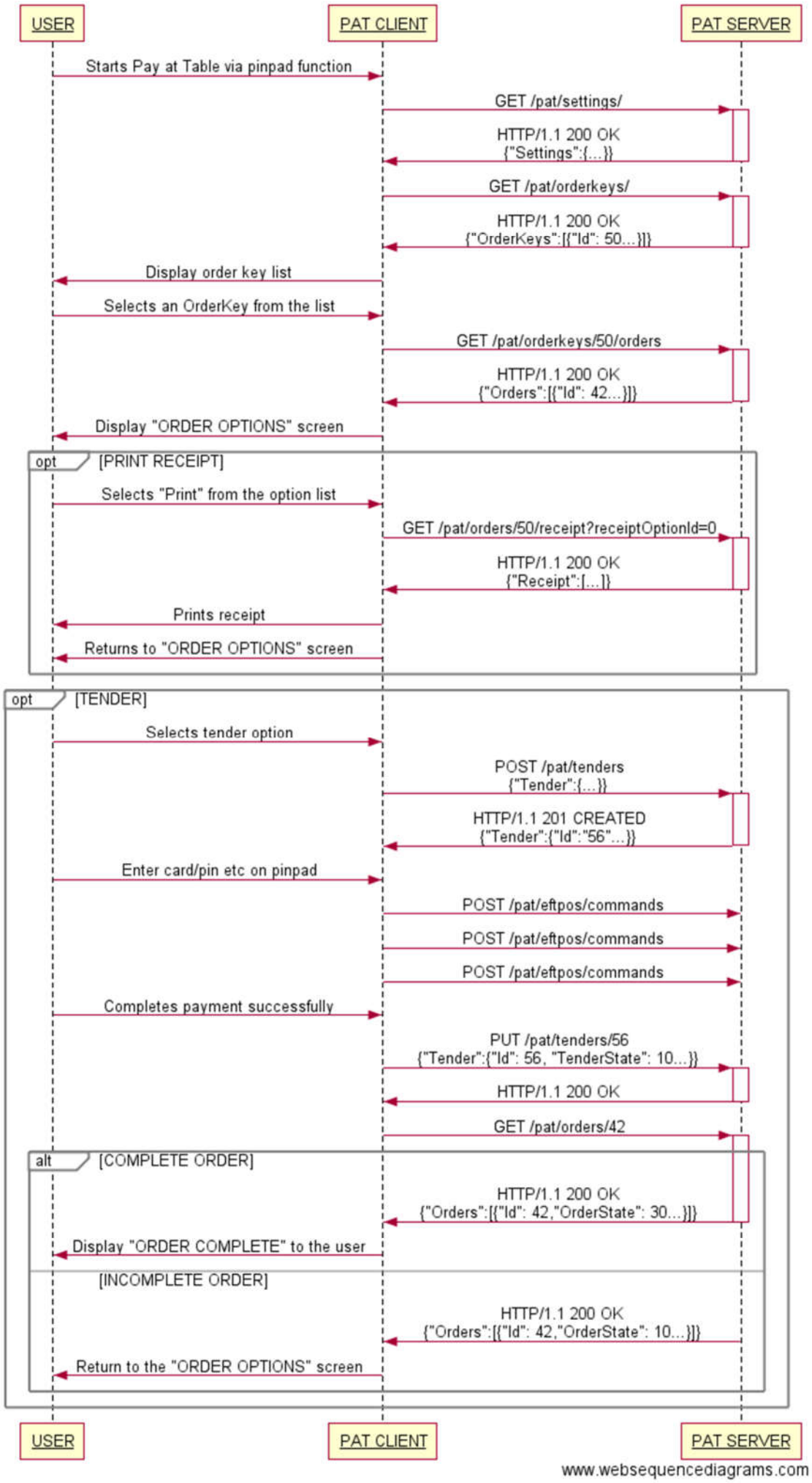
If the POS has implemented the PC-EFTPOS Active X interface the POS client must reside on the same PC as the EFT-Client, if the POS has implemented the PC-EFTPOS TCP/IP interface the POS client can reside on a different PC.



## Example Transaction Flow

- The user initiates a Pay at Table transaction using a configurable function code on the PIN pad
- The Pay at Table client requests the settings from the server
- The Pay at Table client requests a list of tables from the server
- Tables are presented to the user, either as a list using the *DisplayName* property of a *Table* or by allowing the user to manually key a *DisplayNumber*.
- Once the user selects a table, the Pay at Table client requests orders available on that table.
- If no orders are available, the Pay at Table client presents a display to the user and allows them to select another table.
- If orders are available the Pay at Table client presents available options for that order (e.g. print receipt, tender). If multiple orders are available, the Pay at Table client displays all available orders and asks the user to select which order to process.
- If the user selects the *"Print Receipt"* option, the Pay at Table client will request the customer receipt from the server, print it and display the order options again. If multiple print options are available from the settings, the user is asked to select which mode to print before the request is sent to the server.
- If the user selects the *"Tender"* option, the Pay at Table client starts a payment on the PIN pad. If multiple tender options are available from the settings, the Pay at Table client displays these options and asks the user to select the tender type before proceeding with the payment.
- The transaction request, display events and transaction event are sent to the server as EFTPOS commands
- Once the payment is complete, the Pay at Table client updates the tender with a completed state. It is assumed at this point the POS server would also update the order state.
- The Pay at Table client request the selected order again. If the order is complete a message is displayed on the PIN pad, otherwise the user is presented with the order options again.



# Pay at Table Data Request Format

## REST API

### HTTP Response Codes

| HTTP Response Codes | Description |
|---|---|
| **200 OK** | The request was successful |
| **201 Created** | The request was successful and a resource has been created |
| **204 No Content** | The request was successful, there is no content in the response |
| **400 Bad Request** | The client request is invalid |
| **401 Unauthorised** | The client needs to authenticate before it can continue |
| **403 Forbidden** | The client doesn't have access to the resource |
| **404 Not Found** | The requested resource wasn't found |
| **500 Server Error** | The server encountered an internal error processing the request. |

### Methods

| | HTTP Method | Description |
|---|---|---|
| **Get Settings** | GET /api/settings | Get settings for the Pay at Table client |
| **Get Tables** | GET /api/tables | Get a lookup list of tables used to find an order |
| **Get Orders By Table** | GET /api/tables/{table-id}/orders | Get a list of orders associated with a table |
| **Get Order** | GET /api/orders/{order-id} | Get an order based on an order id. |
| **Get Customer Receipt From Order** | GET /api/orders/{order-id}/receipt?receiptOptionId=[string] | Get a customer receipt for a given order. Can accept an optional receipt option id. |
| **Create Tender** | POST /api/tenders | Create a tender |
| **Update Tender** | PUT /api/tenders/{tender-id} | Update a tender |
| **Create EFTPOS Command** | POST /api/eftpos/commands | Create an EFTPOS command |

## Get Settings

### DESCRIPTION

Get the settings for the pay at table client

### REQUEST

```
GET /api/settings
```

Do not supply a request body for this method.

### RESPONSE

If successful the body contains a PATResponse object with the Settings property populated a Settings object.

Supported response codes: 200, 400, 401, 403 and 500.

```
HTTP/1.1 200 OK
Content-type: application/json
{
      "Settings": {
            "TenderOptions": [{
                  "Id": "0",
                  "TenderType": 0,
                  "Merchant": "0",
                  "DisplayName": "EFTPOS",
                  "EnableSplitTender": false,
                  "EnableTipping": true,
                  "CsdReservedString2": "EFTPOS",
                  "TxnType": "P",
                  "PurchaseAnalysisData": ""
            }],
            "ReceiptOptions": [{
                  "Id": "0",
                  "ReceiptType": 0,
                  "DisplayName": "Customer"
            }],
            "PrinterOption": {
                  "PrintMode": 0,
                  "Location": -1,
                  "StaticReceipt": ["line1", "line2", "line3", "line4"]
            }
      }
}
```

## Get Tables

### DESCRIPTION

Get a lookup list of tables used to find an order.

The Pay at Table client will either present a list of selectable items to the user using the “DisplayName” property, or request the user enter a number which will be used to find a table based on the “DisplayNumber” property.

The Id property is a unique identifier for the table used in subsequent requests, and is not presented to the user.

### REQUEST

```
GET /api/tables
```

Do not supply a request body for this method.

### RESPONSE

If successful the body contains a PATResponse object with the Tables property populated by an array of Table.

Supported response codes: 200, 400, 401, 403 and 500.

```
HTTP/1.1 200 OK
Content-type: application/json
{
      "Tables": [{
            "Id": "50",
            "DisplayName": "TABLE 1",
            "DisplayNumber": 1
      },
      {
            "Id": "51",
            "DisplayName": "TABLE 2",
            "DisplayNumber": 2
      },
      {
            "Id": "52",
            "DisplayName": "TABLE 3",
            "DisplayNumber": 3
      }
}
```

## Get Orders by Table

### DESCRIPTION

Get a list of orders associated with a table.

The Pay at Table client will send this request after a user has selected one of the tables returned from a previous call to Get Tables.

### REQUEST

```
GET /api/tables/{table-id}/orders
```

Do not supply a request body for this method

| Parameter | Type | Description |
|---|---|---|
| **table-id** | String | Required. The id of an table orders are being requested from. |

### RESPONSE

If successful the body contains a PATResponse object with the Orders property populated by an array of Order.

Supported response codes: 200, 400, 401, 403 and 500.

```
HTTP/1.1 200 OK
Content-type: application/json
{
      "Orders": [{
            "Id": "101",
            "DisplayName": "Elsa",
            "OrderState": 0,
            "AmountOwing": 100.00,
            "TableId": "50"
      }]
}
```

## Get Order

### DESCRIPTION

Get an order based on an order id.

The Pay at Table client will send this request after a user has selected one of the orders returned from a previous call to Get Orders by Table.

### REQUEST

```
GET /api/orders/{order-id}
```

Do not supply a request body for this method.

| Parameter | Type | Description |
|---|---|---|
| **order-id** | String | Required. The id of the order being requested. |

### RESPONSE

If successful the body contains a PATResponse object with the Order property populated by an Order.

Supported response codes: 200, 400, 401, 403, 404 and 500.

```
HTTP/1.1 200 OK
Content-type: application/json
{
      "Order": {
            "Id": "101",
            "DisplayName": "Elsa",
            "OrderState": 0,
            "AmountOwing": 100.00,
            "TableId": "50"
      }
}
```

## Get Customer Receipt from Order

### DESCRIPTION

Get a customer receipt based on an order id.

### REQUEST

```
GET /api/orders/{order-id}/receipt?receiptOptionId=[string]
```

Do not supply a request body for this method

| Parameter | Type | Description |
|---|---|---|
| **order-id** | String | Required. The id of the order the receipt is being requested from. |
| **receiptOptionId** | String | Optional. The id of the ReceiptOption used to generate this receipt request. Note: id '99' is reserved for custom header/footer receipt from POS |

### RESPONSE

If successful the body contains a PATResponse object with the Receipt property populated by a Receipt.

Supported response codes: 200, 400, 401, 403, 404 and 500.

```
HTTP/1.1 200 OK
{
	"Receipt": {
		"Lines": ["Line 1","Line 2","Line 3"]
	}
}
```

## Create Tender

### DESCRIPTION

Creates a tender. A tender is an object which contains information about a payment.

### REQUEST

```
POST /api/tenders
{
      "Tender": {
            "Id" : null,
            "OrderId": "101",
            "TenderOptionId": "0",
            "TenderState": 0,
            "AmountPurchase": 100.00,
            "OriginalAmountPurchase ": 100.00
      }
}
```

The request body contains a PATRequest with the Tender property populated by a Tender.

The *OrderId* property must reference a valid order.

The *TenderOptionId* property references the tender option selected by the user.

### RESPONSE

If successful the body will contain a PATResponse object with the Tender property populated by a Tender. The Tender in the response will have the Id property populated by a unique Id.

Supported response codes: 201, 400, 401, 403, 404 and 500.

```
HTTP/1.1 201 OK
Content-type: application/json
{
      "Tender": {
            "Id": "1042",
            "OrderId": "101",
            "TenderOptionId": "0",
            "TenderState": 0,
            "AmountPurchase": 100.00,
            "OriginalAmountPurchase": 100.00
      }
}
```

## Update Tender

### DESCRIPTION

Updates a tender.

It is possible that the *AmountPurchase* in an updated tender will not be the same as the *AmountPurchase* in the original tender. E.g. A $100 purchase on a giftcard is completed for the remaining amount on the card ($80.50).

The *Id* property must point to a valid tender and match the {tender-id} in the request url.

The *OrderId* property must point to a valid order.

### REQUEST

```
PUT /api/tenders/{tender-id}
Content-type: application/json
{
      "Tender": {
            "Id": "1042",
            "TenderOptionId": "0",
            "OrderId": "101",
            "TenderState": 2,
            "AmountPurchase": 80.50,
            "OriginalAmountPurchase": 100.00
      }
}
```

The request body contains a PATRequest with the Tender property populated by a Tender.

| Parameter | Type | Description |
|---|---|---|
| **tender-id** | String | Required. The id of the tender being updated. |

### RESPONSE

If successful, this method returns a PATResponse object with the Tender property populated by a Tender. In most cases the Tender in the response will mirror the request.

```
HTTP/1.1 200 OK
Content-type: application/json

{
      "Tender": {
            "Id": "1042",
            "TenderOptionId": "0",
            "OrderId": "101",
            "TenderState": 2,
            "AmountPurchase": 80.50,
            "OriginalAmountPurchase": 100.00
      }
}
```

## Create EFTPOS Command

### DESCRIPTION

Create an EFTPOS command

### REQUEST

The request body contains a PATRequest with the EFTPOSCommand property populated by an EFTPOSCommand

```
POST /api/eftpos/commands
{
      "EFTPOSCommand": {
            "TenderId": "0",
            "OriginalEFTPOSCommandId": "0",
            "EFTPOSCommandType": 0,
            "EFTPOSCommandState": 20
            "AccountType": "",
            "AmtCash": 0.0,
            "AmtPurchase": 100.0,
            "AmtTip": 0.0,
            "AmtTotal": 0.0,
            "Application": "",
            "AuthCode": "",
            "Caid": "",
            "Catid": "",
            "CardName": "",
            "CardType": "",
            "CsdReservedString1": "",
            "CsdReservedString2": "",
            "CsdReservedString3": "",
            "CsdReservedString4": "",
            "CsdReservedString5": "",
            "CsdReservedBool1": false,
            "CutReceipt": false,
            "CurrencyCode": "",
            "DataField": "",
            "Date": "",
            "DateExpiry": "",
            "DateSettlement": "",
            "DialogPosition": "",
            "DialogTitle": "",
            "DialogType": "",
            "DialogX": 0,
            "DialogY": 0,
            "EnableTip": false,
            "EnableTopmost": false,
            "Merchant": "",
```

```
            "MessageType": "",
            "PanSource": " ",
            "Pan": "",
            "PosProductId": "",
            "PurchaseAnalysisData": "",
            "ReceiptAutoPrint": false,
            "ResponseCode": "",
            "ResponseText": "",
            "Rrn": "",
            "Success": false,
            "STAN": "",
            "Time": "",
            "TxnRef": "",
            "TxnType": "",
            "Track1": "",
            "Track2": ""
      }
}
```

### RESPONSE

If successful, this method returns a PATResponse object with the EFTPOSCommand property populated by an EFTPOSCommand. In most cases the EFTPOSCommand in the response will mirror the request.

```
HTTP/1.1 200 OK
Content-type: application/json
{
      "EFTPOSCommand": {
            "TenderId": "0",
            "OriginalEFTPOSCommandId": "0",
            "EFTPOSCommandType": 0,
            "EFTPOSCommandState": 20
            "AccountType": "",
            "AmtCash": 0.0,
            "AmtPurchase": 100.0,
            "AmtTip": 0.0,
            "AmtTotal": 0.0,
            "Application": "",
            "AuthCode": "",
            "Caid": "",
            "Catid": "",
            "CardName": "",
            "CardType": "",
            "CsdReservedString1": "",
            "CsdReservedString2": "",
            "CsdReservedString3": "",
```

```
            "CsdReservedString4": "",
            "CsdReservedString5": "",
            "CsdReservedBool1": false,
            "CutReceipt": false,
            "CurrencyCode": "",
            "DataField": "",
            "Date": "",
            "DateExpiry": "",
            "DateSettlement": "",
            "DialogPosition": "",
            "DialogTitle": "",
            "DialogType": "",
            "DialogX": 0,
            "DialogY": 0,
            "EnableTip": false,
            "EnableTopmost": false,
            "Merchant": "",
            "MessageType": "",
            "PanSource": " ",
            "Pan": "",
            "PosProductId": "",
            "PurchaseAnalysisData": "",
            "ReceiptAutoPrint": false,
            "ResponseCode": "",
            "ResponseText": "",
            "Rrn": "",
            "Success": false,
            "STAN": "",
            "Time": "",
            "TxnRef": "",
            "TxnType": "",
            "Track1": "",
            "Track2": ""
      }
}
```

# PC-EFTPOS Interface

When using the existing PC-EFTPOS interface, the Pay at Table client uses a similar request/response structure to the REST server, but wrapped in the existing PC-EFTPOS interface.

## ActiveX Interface

### POS to EFT-Client command

The POS to EFT-Client command is used by a POS responding to a request for data by the Pay at Table client.

- Set TxnType to '@'
- Set CsdReservedString1 to the Pay at Table response structure below
- Call DoCsdReserved3()

| # | Field | Length | Format | Description |
|---|---|---|---|---|
| **1** | Header length | 6 | Numeric. Right aligned, zero padded. | The length of the Pay at Table header to follow. |
| **2** | Response header | * | Alphanumeric | JSON formatted Pay at Table response header. To find the length of the content, check Content-length in the header. |
| **3** | Response content | * | Alphanumeric | JSON formatted Pay at Table response content |

### EFT-Client to POS command

The EFT-Client to POS command is used by the Pay at Table client to request data from the POS.

- OnCsdReserved3 event will fire
- TxnType will be set to '@'
- DataField will be set to the Pay at Table request structure below

| # | Field | Length | Format | Description |
|---|---|---|---|---|
| **1** | Header length | 6 | Numeric. Right aligned, zero padded. | The length of the Pay at Table header to follow. |
| **2** | Request header | * | Alphanumeric | JSON formatted Pay at Table request header. To find the length of the content, check Content-length in the header. |
| **3** | Request content | * | Alphanumeric | JSON formatted Pay at Table request content |

## Pay at Table request header

The request header is JSON formatted.

| Field | Format | Description |
|---|---|---|
| **Version** | Numeric | Pay at Table message version. Default to 1 |
| **Content-type** | Alphanumeric | application/json |
| **Request-type** | Alphanumeric | GET,PUT,POST,DELETE |
| **Request-method** | Alphanumeric | Settings<br>Tables<br>TableOrders<br>Order<br>OrderReceipt<br>Tender<br>EFTPOSCommand |
| **Content-length** | Numeric | The length of the content to follow |

## Pay at Table response header

The header is JSON formatted.

| Field | Format | Description |
|---|---|---|
| **Version** | Numeric | Pay at Table message version. By default, 1. |
| **Content-type** | Alphanumeric | application/json |
| **Request-type** | Alphanumeric | Mirrored from the request |
| **Request-method** | Alphanumeric | Mirrored from the request |
| **Response-code** | Numeric | One of the HTTP response codes. 200, 201, 204, 400, 401, 403, 404, 500. |
| **Response-text** | Alphanumeric | One of the HTTP response codes texts |
| **Content-length** | Numeric | The length of the content to follow |

## Sample

Get Tables method

Request [HeaderLength][Header][Content]

A length of 137, followed by the request header. There is no content so that field is not included.

```
000137
{
	"Version": 1,
	"ContentType": "application/json",
	"RequestType": "GET",
	"RequestMethod": "Tables",
	"ContentLength": 0
}
```

Response [HeaderLength][Header][Content]

```
000188
{
	"Version": 1,
	"ContentType": "application/json",
	"RequestType": "GET",
	"RequestMethod": "Tables",
	"ResponseCode": 200,
	"ResponseText": "OK",
	"ContentLength": 0
}
{
	"Tables": [{
		"Id": "50",
		"DisplayName": "TABLE 1",
		"DisplayNumber": 1
	},
	{
		"Id": "51",
		"DisplayName": "TABLE 2",
		"DisplayNumber": 2
	}
}
```

## TCP/IP Interface

TCP/IP Interface

### POS to EFT-Client command

- Construct a EFTPayAtTableRequest object specifying the Response Header and Content in JSON format as defined in the table below.
- Call WriteRequestAsync supplying the EFTPayAtTableRequest as the parameter.

| # | Field | Length | Format | Description |
|---|---|---|---|---|
| **1** | Start flag | 1 | Alphanumeric | Content header. Default to '#' |
| **2** | Command code | 1 | Alphanumeric | Generic POS command type. Default to 'X' |
| **3** | Sub code | 1 | Alphanumeric | Pay at Table command type. Default to '@' |
| **4** | Header length | 6 | Numeric. Right aligned, zero padded. | The length of the Pay at Table header to follow. |
| **5** | Response Header | * | Alphanumeric | JSON formatted header. To find the length of the content, check Content-length in the header. |
| **6** | Content | * | Alphanumeric | JSON formatted Pay at Table response content |

### EFT-Client to POS command

- Call ReadResponseAsync.
- Await a returned object type of EFTPayAtTableResponse to access the Request Header and Content.

| # | Field | Length | Format | Description |
|---|---|---|---|---|
| **1** | Start flag | 1 | Alphanumeric | Content header. Default to '#' |
| **2** | Command code | 1 | Alphanumeric | Generic POS command type. Default to 'X' |
| **3** | Sub code | 1 | Alphanumeric | Pay at Table command type. Default to '@' |
| **4** | Header length | 6 | Numeric. Right aligned, zero padded. | The length of the Pay at Table header to follow. |
| **5** | Request Header | * | Alphanumeric | JSON formatted Pay at Table request header. To find the length of the content, check Content-length in the header. |
| **6** | Content | * | Alphanumeric | JSON formatted Pay at Table request content |

### Sample

GET Settings method

Request

[Start flag][message length][Command Code][Sub-code][Response Message][Header Length][Header][Content]

A length of 159, followed by the request header. There is no content so that field is not included.

```
#0192X@APPROVED            000159{
  "Version": 1,
  "ContentType": "application/json",
  "RequestType": "GET",
  "RequestMethod": "Settings",
  "ContentLength": 0,
  "TableID": "",
```

```
  "OrderID": "",
  "ReceiptOptionId": ""
}
```

Response

[Start flag][Message length][Command Code][Sub-code][header Length][Header][Content]

```
#0619X@000323{
  "Version": 1,
  "ContentType": "application/json",
  "RequestType": "GET",
  "RequestMethod": "Settings",
  "ContentLength": 283,
  "TableId": "",
  "OrderId": "",
  "ReceiptOptionId": "",
  "tender": {
    "Id": null,
    "OrderId": null,
    "TenderState": 0,
    "TenderOptionId": null,
    "AmountPurchase": 0.0,
    "OriginalAmountPurchase": 0.0
  },
  "ResponseCode": 200,
  "ResponseText": "Ok"
}
{
  "Tables": null,
  "Orders": null,
  "Order": null,
  "Receipt": null,
  "EFTPOSCommand": null,
  "Tender": null,
  "Settings": {
    "TenderOptions": [
      {
        "Id": "",
        "TenderType": 0,
        "Merchant": "00",
        "DisplayName": "EFTPOS",
        "EnableSplitTender": false
      }
    ],
    "ReceiptOptions": [
      {
        "Id": "",
        "ReceiptType": 0,
        "DisplayName": "Customer"
      }
    ]
  }
}
```

# Model

## PATRequest

### DESCRIPTION

A wrapper for a request to the pay at table API. The contents will depend on the method being called.

```
{
	"EFTPOSCommand": ...,
	"Tender": ...
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **EFTPOSCommand** | EFTPOSCommand | Represent EFT-client request commands |
| **Tender** | Tender | Represents a payment |

## PATResponse

### DESCRIPTION

A wrapper for a response from the pay at table API. The contents will depend on the method which generated the response.

```
{
	"Tables": [],
	"Orders": [],
	"Order": ...,
	"Receipt": ...,
	"EFTPOSCommand": ...,
	"Tender": ...,
	"Settings": ...
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **EFTPOSCommand** | EFTPOSCommand | Represents EFT-client request commands |
| **Tender** | Tender | Represents a payment |
| **Orders** | Order[] | An array of Order |
| **Tables** | Table[] | An array of Table |
| **Order** | Order | Represents a sale |
| **Receipt** | Receipt | Proof of sale |
| **Settings** | Settings | Defines settings for the pay at table client |

## TenderOption

### DESCRIPTION

The tender option describes a payment option available to the Pay at Table client. This will typically be “EFTPOS”, however other options (such as gift card) could be supported.

```
{
	"Id": "0",
	"TenderType": 0,
	"Merchant": "0",
	"DisplayName": "EFTPOS",
	"EnableSplitTender": false,
	"EnableTipping": true,
	"CsdReservedString2": "EFTPOS",
	"TxnType": "P",
	"PurchaseAnalysisData": ""
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Id** | String | A unique identifier for this tender option. This is passed back to the server when a tender is created. |
| **TenderType** | Integer | Defines how this tender option is handled by the Pay at Table client.<br><br>Possible values:<br>• (0) EFTPOS |
| **Merchant** | String | The merchant code to use in the request if this tender option is to be sent to a PIN pad. Default to “00”. |
| **DisplayName** | String | Max 14 characters. A name which can be presented to the user to identify this tender option |
| **CsdReservedString2** | String | This property defines which application the EFT-Client is to send the transaction details to. If the property is empty, the default EFTPOS application will be used. Other possible values:<br>• “EFTPOS” - Use the EFTPOS application (default)<br>• “AGENCY” - Use the Agency application within the terminal. |
| **TxnType** | String | 1 character text property that determines the type of transaction to perform. If empty, the default “P” is sent out. Possible values:<br>• “P” – Purchase Cash<br>• “R” – Refund<br>• etc. |
| **EnableTipping** | Boolean | Indicates to the PC-EFTPOS system to perform a purchase with a possible tip. |
| **EnableSplitTender** | Boolean | True if the user should be able to tender for an amount less than the total of the order. If false, the user will not be able to change the amount displayed in the PIN pad. |
| **PurchaseAnalysisData** | String | Sent down to the PIN pad in the PurchaseAnalysisData field |

## ReceiptOption

### DESCRIPTION

Describes a receipt option available to the Pay at Table client. This will typically be “Customer”, however other options could be supported.

```
{
	"Id": "0",
	"ReceiptType": 0,
	"DisplayName": "Customer"
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Id** | String | A unique identifier for this receipt option. This is passed back to the server when a receipt is requested.<br>Note: ‘99’ is reserved for custom POS header/footer receipts |
| **ReceiptType** | Integer | Defines the receipt type.<br><br>Possible values:<br>• (0) Order |
| **DisplayName** | String | Max 14 characters. A name which can be presented to the user to identify this receipt option |

## Settings

### DESCRIPTION

Defines settings for the pay at table client.

```
{
	"Settings": {
		"TenderOptions": [{
			"Id": "0",
			"TenderType": 0,
			"Merchant": "0",
			"DisplayName": "EFTPOS",
			"EnableSplitTender": false,
			"EnableTipping": true,
			"CsdReservedString2": "EFTPOS",
			"TxnType": "P",
			"PurchaseAnalysisData": ""
		}],
		"ReceiptOptions": [{
			"Id": "0",
			"ReceiptType": 0,
			"DisplayName": "Customer"
		}],
		"PrinterOption": {
			"PrintMode": 0,
			"Location": -1,
			"StaticReceipt": ["line1", "line2", "line3", "line4"]
		}
	}
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **TenderOptions** | TenderOption[] | Lists the tender options available to the Pay at Table client.<br>If left null or empty the option to tender will not be available on Pay at Table client when the user selects an order.<br>If only one option is available, the Pay at Table client will automatically select that option when the user chooses to tender. |
| **ReceiptOptions** | ReceiptOption[] | Lists the tender options available to the Pay at Table client.<br>If left null or empty the option to print will not be available on Pay at Table client when the user selects an order.<br>If only one option is available, the Pay at Table client will automatically select that option when the user chooses an order. |
| **PrinterOption** | PrinterOption | Allows for custom receipt information, appended to the header/footer of the eftpos receipt, if left null no custom receipt will be printed.<br>You can allow PC-EFTPOS to print a custom header/footer with some information about the current Transaction<br>Defaults to 'PCEFTPOS' is printing and 'No printing' |

## Table

### DESCRIPTION

Defines an item in a lookup table used to find an order.

For example, an array of Table could represent the tables in a restaurant. The user could then be presented with either a list of table names contained in the *DisplayName* property, or the *DisplayNumber* property could be used to select a specific *Table*.

After the user has selected an *Table*, the Pay at Table client will call the */api/tables/{table-id}/orders* method to retrieve the orders available for this *Table*.

```
{
	"Id": "50",
	"DisplayName": "TABLE 1",
	"DisplayNumber": 1
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Id** | String | Unique identifier. |
| **DisplayName** | String | Max 14 characters. A name which represents this table that could be displayed to a user. |
| **DisplayNumber** | Integer | A number which represents this table that could be displayed to a user. |

## Order

### DESCRIPTION

An *Order* defines a sale. Orders available for tender will have an *OrderState* set to 10 (active).

```
{
	"Id": "101",
	"DisplayName": "Elsa",
	"OrderState": 0,
	"AmountOwing": 100.00,
	"TableId": "50"
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Id** | String | Unique identifier. Read only. |
| **DisplayName** | String | Max 14 characters. A name which represents this table that could be displayed to a user. |
| **OrderState** | Integer | The state of the order. This is used by the Pay at Table client to determine if an order is available for tender.<br>Possible values:<br>• (0) Pending – The order exists, but isn't yet available for tender.<br>• (10) Active – The order exists and is available for tender.<br>• (20) Tendering – A tender is currently in progress. The result is not known. The order is not available for tender.<br>• (30) Complete – The order is complete and is not available for tender. |
| **AmountOwing** | Decimal | The outstanding amount on this order. This is used by the Pay at Table client to determine the maximum tender amount. |
| **TableId** | String | The id of the *Table* attached to this order. Can be null. |

## Tender

### DESCRIPTION

A *Tender* defines a payment.

```
{
      "Id": "1042",
      "OrderId": "123"
      "TenderOptionId": "0",
      "TenderState": 2,
      "AmountPurchase": 80.00,
      "OriginalAmountPurchase": 100.00,
      "EFTResponseCode": "00",
      "EFTResponseText": "APPROVED"
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Id** | String | Unique identifier. Read only. |
| **TenderOptionId** | String | The id of the tender option the operator selected to create this tender |
| **TenderState** | Integer | The state of a tender is defined by the *TenderState* property. The initial state is set to *Pending* (0). When the payment is complete the *Tender* object will be updated and the *TenderState* changed to *CompletedSuccessful* (1) or *CompletedUnsuccessful* (2). |
| **AmountPurchase** | Decimal | The amount of this tender (see notes) |
| **Original AmountPurchase** | Decimal | If the tender amount is changed (e.g. A $100 purchase on a gift card is completed for the remaining amount on the card - $80.50) this value will reflect the original tender amount before it was changed. |
| **OrderId** | String | The id of the order this tender is attached to. Can be NULL. |
| **EFT ResponseCode** | String | The response code returned from the PIN pad |
| **EFT ResponseText** | String | The response text returned from the PIN pad |

### NOTES

Although not applicable to EFTPOS or credit card sales, in some instances (such as a partial gift card payment) the AmountPurchase can potentially change between the original “create tender” request and the “update tender” request.

For example, the operator selects a gift card tender option with a gift card provider that supports partial tenders. A sale of $100 is tendered, however there is only $20 available on the gift card so the $100 tender is approved for $20.

In this example, the “create tender” request would have an amount of $100, and the subsequent “update tender” request would have the amount changed to $20 with the OriginalAmountPurchase set to $100.

## Receipt

### DESCRIPTION

```
{
	"Receipt":
	{
		"Lines":
		[
			"---------LINE 1---------",
			"---------LINE 2---------",
			"---------LINE 3---------"
		]
	}
}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Lines** | String[] | An array of lines to appear on the receipt.<br><br>Each receipt line has a maximum of 24 characters |

## PrinterOption

### DESCRIPTION

Optional. Custom header/footer receipts appended to the eftpos receipt.

```
{
      "PrinterOption": {
                  "PrintMode": 0,
                  "Location": -1,
                  "StaticReceipt": ["line1", "line2", "line3", "line4"]
            }

}
```

### PROPERTIES

| Name | Type | Description |
|---|---|---|
| **PrintMode** | Int | Who handles the printing of custom header/footer receipts.<br>0 = PCEFTPOS<br>1 = POS<br>2 = Static<br>0. You can have PC-ETPOS print a default custom receipt with some information about the current transaction (Table, order#, Tender amount…)<br>1. The POS can implement their own custom header/footer by using this mode, you will need to implement a receipt for ReceiptOptionId '99', this will be used as the POS custom header/footer.<br>2. This option allows for a static header/footer to be printed such as 'thank you for shopping with us' which is defined in the 'StaticReceipt' property. |
| **Location** | Int | Where to display the custom receipt<br>0 = Header(before eftpos receipt)<br>1 = Footer (after eftpos receipt)<br>-1 = None(No custom receipt to be printed)<br>0. This will print the custom receipt in the header, before the eftpos receipt is printed.<br>1. This will print the custom receipt in the footer, after the eftpos receipt is printed and the transaction is complete(operators will need to wait until 'Order Complete' is displayed on the eftpos terminal before cutting the receipt)<br>-1. Disables printing of custom receipts, this is the default value. |
| **StaticReceipt** | String[] | An array of lines to appear on the custom receipt. ONLY works if PrintMode is set to '2'.<br><br>Each receipt line has a maximum of 24 characters. |

## EFTPOSCommand

Represent EFT-client request commands

### DESCRIPTION


```
      "EFTPOSCommand": {
            "TenderId": "0",
            "OriginalEFTPOSCommandId": "0",
            "EFTPOSCommandType": 0,
            "EFTPOSCommandState": 20
            "AccountType": "",
            "AmtCash": 0.0,
            "AmtPurchase": 100.0,
            "AmtTip": 0.0,
            "AmtTotal": 0.0,
            "Application": "",
            "AuthCode": "",
            "Caid": "",
            "Catid": "",
            "CardName": "",
            "CardType": "",
            "CsdReservedString1": "",
            "CsdReservedString2": "",
            "CsdReservedString3": "",
            "CsdReservedString4": "",
            "CsdReservedString5": "",
            "CsdReservedBool1": false,
            "CutReceipt": false,
            "CurrencyCode": "",
            "DataField": "",
            "Date": "",
            "DateExpiry": "",
            "DateSettlement": "",
            "DialogPosition": "",
            "DialogTitle": "",
            "DialogType": "",
            "DialogX": 0,
            "DialogY": 0,
            "EnableTip": false,
            "EnableTopmost": false,
            "Merchant": "",
            "MessageType": "",
            "PanSource": " ",
            "Pan": "",
```

```
            "PosProductId": "",
            "PurchaseAnalysisData": "",
            "ReceiptAutoPrint": false,
            "ResponseCode": "",
            "ResponseText": "",
            "Rrn": "",
            "Success": false,
            "STAN": "",
            "Time": "",
            "TxnRef": "",
            "TxnType": "",
            "Track1": "",
            "Track2": ""
    }
```

## PROPERTIES

| Name | Type | Description |
|---|---|---|
| **Id** | String | Unique identifier. Read only. |
| **TenderId** | String | Id of the tender that this EFTPOS command is associated with |
| **OriginalEFTPOS CommandId** | String | The id of the original EFTPOS request if this is an event. |
| **EFTPOSCommand Type** | Integer | DoTransaction = 100, DoLogon = 101, TransactionEvent = 200, LogonEvent = 201, DoKeyPress = 300, DisplayEvent = 400, PrintEvent = 401 |
| **EFTPOSCommand State** | Integer | AwaitingDeviceAck = 0, AwaitingDeviceResponse = 10, CompletedSuccessful = 20, CompletedUnsuccessful = 30 |
| **AccountType** | String | |
| **AmtCash** | Decimal | |
| **AmtPurchase** | Decimal | |
| **AmtTip** | Decimal | |
| **AmtTotal** | Decimal | |
| **Application** | String | |
| **AuthCode** | String | |
| **Caid** | String | |
| **Catid** | String | |
| **CardName** | String | |
| **CardType** | String | |
| **CsdReservedString1** | String | |
| **CsdReservedString2** | String | |
| **CsdReservedString3** | String | |
| **CsdReservedString4** | String | |
| **CsdReservedString5** | String | |
| **CsdReservedBool1** | Boolean | |
| **CutReceipt** | Boolean | |
| **CurrencyCode** | String | |
| **DataField** | String | |
| **Date** | String | |
| **DateExpiry** | String | |
| **DateSettlement** | String | |
| **DialogPosition** | String | |
| **DialogTitle** | String | |
| **DialogType** | String | |

| | | |
|---|---|---|
| **DialogX** | Integer | |
| **DialogY** | Integer | |
| **EnableTip** | Boolean | |
| **EnableTopmost** | Boolean | |
| **Merchant** | String | |
| **MessageType** | String | |
| **PanSource** | String | |
| **Pan** | String | |
| **PosProductId** | String | |
| **PurchaseAnalysisData** | String | |
| **ReceiptAutoPrint** | Boolean | |
| **ResponseCode** | String | |
| **ResponseText** | String | |
| **Rrn** | String | |
| **Success** | Boolean | |
| **STAN** | String | |
| **Time** | String | |
| **TxnRef** | String | |
| **TxnType** | String | |
| **Track1** | String | |
| **Track2** | String | |