# Secure Web Pay Checkout

## Integration Guide v1.10

forte
a CSG Company

# Revision History

| Document Version | Date | Changes |
|---|---|---|
| 1.0 | 06/01/2015 | Initial |
| 1.1 | 07/02/2015 | Added note explaining disparities or response hash signatures between version 1.0 and version `2.0` of SWP Checkout. |
| 1.2 | 09/02/2015 | Added the "Using Postback Information" section to the SWP Checkout chapter. |
| 1.3 | 09/25/2015 | Updated character length for the following fields:<br>• `pg_consumerorderid`<br>• `pg_walletid`<br>• `pg_merchant_data_1-4` |
| 1.4 | 11/16/2016 | Instances of `pg_return_URL` changed to `pg_return_url` for consistency. |
| 1.5 | 02/27/2017 | • Removed references to CMI.<br>• Updated the live URL for the SWP Embedded Capture Template<br>• Updated the hash parameter to `TSHash`. |
| 1.6 | 08/16/2016 | • Removed references to SWP Embedded Charge Template, Embedded Capture Template, and SWP Redirect.<br>• New layout to accommodate just the SWP Checkout section. |
| 1.7 | 12/13/2017 | • Added the formula for validating the `pg_ts_hash_response` parameter in postbacks.<br>• Updated list of potential postback parameters |
| 1.8 | 07/24/2018 | Added the `pg_receipt` parameter |
| 1.9 | 10/21/2020 | Added the `pg_entry_class_code` parameter. |
| 1.10 | 05/30/2021 | Added `U91` and `U92` response codes. |

# Table of Contents



 

# Introduction

## Overview

Forte's Secure Web Pay (SWP) Checkout
- Captures purchase information via swipe or key entry
- Processes credit card, EFT, and recurring transactions
- Automatically responds to your point-of-sale machine approving or denying the transaction

SWP Checkout is easy to integrate with your browser-based applications. It works by calling a customizable Forte-hosted payment form where customers can enter and securely submit both ad-hoc and recurring transaction information. SWP Checkout includes customer authentication, tokenization support, and is PCI-DSS compliant.

## Advantages of SWP Checkout

SWP Checkout enables web merchants to call a Forte-hosted payment form to capture and securely submit transaction information. Merchants can customize the look and feel of the hosted payment form by selecting either a predefined payment form style or passing in configuration fields with the call to the payment form.

With a SWP Checkout integration, merchants
- Do not have to maintain a Secure Sockets Layer (SSL) certificate
- Reduce PCI scope and fraud liability
- Enjoy enhanced security through Forte's PCI and NACHA audits

# Understanding Checkout Parameters

The following table displays the types of parameters merchants can use in the SWP Checkout application. This table should be used in conjunction with the request and response parameter tables displayed in the following sections.

## Parameter Type Definitions

| Type | Description | Characters Allowed | Case Sensitive? |
|---|---|---|---|
| M | Money | 0–9 (and an optional period) | N/A |
| N | Numeric | 0–9 (no period allowed) | N/A |
| A | Alphanumeric | Any printable ASCII | Yes |
| L | List-Based Value | Value must be in the specified list | No |
| D | Date | Format: DD/MM/YYYY | N/A |
| T | True/False | True or False only | No |

## Request Parameters

Forte uses specific parameters to populate values on the SWP Checkout payment page as read-only. Prefixing a parameter with an `e_` will allow the values to be shown as editable.

| Name | Description | Req | Type |
|---|---|---|---|
| **Bill To** | | | |
| `pg_billto_postal_name_company` | Company Name | O | A20 |
| `pg_billto_postal_name_first` | First Name | R | A25 |
| `pg_billto_postal_name_last` | Last Name | R | A25 |
| `pg_billto_postal_street_line1` | Address1 | O | A35 |
| `pg_billto_postal_street_line2` | Address2 | O | A35 |
| `pg_billto_postal_city` | City | O | A25 |
| `pg_billto_postal_stateprov` | State | O | A10 |
| `pg_billto_postal_postalcode` | Postal Code | O | A10 |
| `pg_billto_telecom_phone_number` | Phone Number (*nnn-nnn-nnnn* or *nnnnnnnnnn*) | O | A15 |
| `pg_billto_online_email` | Email address | O | A40 |
| **Ship To** | | | |
| `pg_shipto_postal_name` | Ship to Name | O | A35 |
| `pg_shipto_postal_street_line1` | Ship to Address 1 | O | A35 |
| `pg_shipto_postal_street_line2` | Ship to Address 2 | O | A35 |
| `pg_shipto_postal_city` | Ship to City | O | A25 |
| `pg_shipto_postal_stateprov` | Ship to State | O | A10 |
| `pg_shipto_postal_postalcode` | Ship to Zip Code | O | A10 |

*Continued on next page*

# Understanding Checkout Parameters, Continued

**Request Parameters**, continued

| Name | Description | Req | Type |
|---|---|---|---|
| **Order Information** | | | |
| pg_consumer_id | This field identifies the customer and can be searched for in Virtual Terminal. This is an API-only field and is not shown on the Payments page. | O | A15 |
| pg_consumerorderid | Invoice number of the transaction. Shows up in Virtual Terminal as Consumer Order ID or your labelled field. This field should be numeric for First Data customers to prevent downgrading. Set in Virtual Terminal as **Optional** or **Required** to use. | O | A36 |
| pg_wallet_id | Description of the transaction. Shows up in Virtual Terminal as **Wallet ID** or as your labelled field. Set in Virtual Terminal as **Optional** or **Required** to use. | O | A15 |
| pg_merchant_data_1 | Optional Field 1. This field is off by default. Set in Virtual Terminal as **Optional** or **Required** to use. | O | A255 |
| pg_merchant_data_2 | Optional Field 2. This field is off by default. Set in Virtual Terminal as **Optional** or **Required** to use. | O | A255 |
| pg_merchant_data_3 | Optional Field 3. This field is off by default. Set in Virtual Terminal as **Optional** or **Required** to use. | O | A255 |
| pg_merchant_data_4 | Optional Field 4. This field is off by default. Set in Virtual Terminal as **Optional** or **Required** to use. | O | A255 |
| pg_line_item_header | Line items header. Example: name, quantity, price, etc. | O | A8000 |
| pg_line_item_1 | Line item 1–100. Example: alpha, 1, 1.00 | O | — |
| pg_sales_tax_amount | Sales tax amount. This is an API-only field. It is a read-only field on the Payments page. | O | M |
| pg_entry_class_code | Use one of the following values if you want to pass an echeck transaction with another SEC code besides the default value WEB:<br>• ARC<br>• BOC<br>• CCD<br>• CIE<br>• CTX<br>• POP<br>• POS<br>• PPD<br>• RCK<br>• TEL<br><br>**NOTE:** Your merchant configuration must be set up to support the SEC code you select. | O | A3 |

*Continued on next page*

# Understanding Checkout Parameters, Continued

**Request Parameters**, continued

| Name | Description | Req | Type |
|---|---|---|---|
| **Recurring** | | | |
| pg_scheduled_transaction | 0 = not a scheduled transaction<br>1 = scheduled transaction | O | N1 |
| pg_schedule_quantity | Quantity of transactions to be performed. Set to 0 for a single future transaction. This parameter is required for scheduled transactions. | R | N9 |
| pg_schedule_frequency | 10 = Weekly<br>15 = Bi-Weekly (Every 14 days)<br>20 = Monthly<br>25 = Bi-Monthly (Every 2 months)<br>30 = Quarterly (Every 3 months)<br>35 = Semi-Annually<br>40 = Annually<br>Set to 0 for a single future transaction. This parameter is required for scheduled transactions. | R | L |
| pg_schedule_start_date | Specifies the start date of the next recurring transaction (*MM/DD/YYYY*). Today or greater. This parameter is required for scheduled transactions. | R | D |
| pg_schedule_continuous | 0 = Not continuous<br>1 = Continuous<br>Set to 0 for single future transaction. | O | N1 |
| **Authentication** | | | |
| pg_api_login_id | API Login ID in the Virtual Terminal | R | |
| pg_transaction_type | 10 = Credit Card Sale<br>20 = eCheck Sale<br>11 = Credit Card Authorization<br>21 = eCheck Authorization<br>This field is optional for HTML integrations. If it is not sent, the customer will have both sale options available depending on the permissions in the Virtual Terminal. This parameter is required for signed transactions. | R | L |
| pg_version_number | Current version 1.0. This parameter is required for signed transactions. | R | A3 |
| pg_total_amount | Total amount of transaction to be charged/credited to the customer. This parameter is required for signed transactions. | R | |
| pg_utc_time | UTC time in ticks (since 01/01/0001 00:00:00). This parameter is required for signed transactions. | R | |

*Continued on next page*

# Understanding Checkout Parameters, Continued

**Request Parameters**, continued

| Name | Description | Req | Type |
|---|---|---|---|
| **Authentication** | | | |
| pg_transaction_order_number | Random number identifying the transaction. This parameter is required for signed transactions. | R | |
| pg_ts_hash | The hash generated by the transaction signing fields. This parameter is required for signed transactions. | R | |
| **Other** | | | |
| pg_return_url | The page the customer will be returned to after a transaction has completed. This page should contain a server-side script to parse the data being posted to it; however, it can be a static HTML page as well. If this field is not set or is invalid, the entire transaction is completed on Forte's side. The URL sent must match at least one of the URLs set in Virtual Terminal. | O | A100 |
| pg_continue_url | After any payment has been completed (Pass or Fail), this will be the URL the customer is directed to when he/she presses the **Return** button. | O | A100 |
| pg_continue_description | The text of the button used for the **Continue URL**. If not passed, the default value is **Return**. | O | A25 |
| pg_return_method | The return method for the postback. Use **AsyncPost** for reliability and security. The AsyncPost method does not rely on the user's browser to perform a postback. | O | A10 |
| pg_cancel_url | The page the customer will be returned to after the customer selects the **Abort** button. The **Abort** button is shown on failed transactions. The default value is **forte.net**. | O | A100 |
| pg_save_client | 1 = Create payment method token<br>2 = Create client and payment method tokens<br><br>If the merchant does not pass this field, tokens are not created for this transaction. | O | N |

*Continued on next page*

# Understanding Checkout Parameters, Continued

**Request Parameters**, continued

| Name | Description | Req | Type |
|---|---|---|---|
| pg_customer_ip_address | The ip address from which the transaction originated. | O | A80 |
| pg_swipe | Supported values equal **True** or **False**. If **True**, swipers are enabled.<br><br>Forte supports Magtek model #30050200 and #21073062. Swipers must be purchased from Forte.<br><br>If **True**, two buttons will appear on the payment form. The **Keyed Entry** button allows for manual entry via a keyboard. The **Swipe** button allows for both swiping the credit card and using the IPAD PIN pad. | O | T |
| pg_tc_show | Supported values equal **True** or **False**. If **False**, no Terms and Conditions are displayed. The default value is **True**. | O | T |
| pg_signature_show | Supported values equal **True** or **False**. If **True**, a signature line will show on the receipt. The default value is **False**. | O | T |
| pg_receipt | Defines the type of receipt you want to generate for successful transactions. Supported options include the following:<br>• 1 = 8.5-inch receipt<br>• 2 = 3-inch receipt | O | N1 |

**Recurring Transactions**

The recurring fields of the Recurring Transaction Template are used to establish recurring, scheduled transactions. Transactions will be created and processed at the stated frequency (as long as the recurring transaction is in an "active" state). The transactions will be created and processed until the specified quantity is reached (if it is non-zero) or until the transaction is suspended or deleted by the merchant. Voided and declined transactions do not count towards the specific quantity. Recurring transactions must have both pg_scheduled_quantity and pg_schedule_frequency, but the pg_schedule_start_date parameter is optional.

# Understanding Response Parameters

## Overview

Depending on the data passed in the request, SWP Checkout posts the following data to the `pg_return_url` page:

- Billing and shipping address data
- Up to four merchant data fields, if passed
- `pg_consumerorderid` and `pg_wallet_id` – the invoice number and transaction description for easy transaction tracking
- Last four digits of the credit card or echeck account number
- The client token (i.e., `pg_client_id`) and the payment method token (i.e., `pg_payment_method_id`) if the merchant sends `pg_save_client=2` in the request or just the payment method token if the merchant sends `pg_save_client=1` in the request
- `pg_response_code` – The transaction response code. See below.

# Understanding Response Parameters, Continued

, continued

| Code | Description | Comments | Test Parameters |
|---|---|---|---|
| U03 | DAILY TRANS LIMIT | Merchant daily limit exceeded (EFT only) | Not available |
| U04 | MONTHLY TRANS LIMIT | Merchant monthly limit exceeded (EFT only) | Not available |
| U05 | AVS FAILURE ZIP CODE | AVS State/Zip Code check failed | Set `pg_avs_method=00200` and send a state and zip code that do not match. |
| U06 | AVS FAILURE AREA CODE | AVS State/Area Code check failed | Set `pg_avs_method=00200` and send a state and area code that do not match. |
| U07 | AVS FAILURE EMAIL | AVS anonymous email check failed | Set `pg_avs_method=00200` and send an email for Hotmail.com. |
| U10 | DUPLICATE TRANSACTION | Transaction has the same attributes as another transaction within the time set by the merchant. | Send the same transaction twice within five minutes. |
| U11 | RECUR TRANS NOT FOUND | Transaction Types `40–42` only | Not available |
| U12 | UPDATE NOT ALLOWED | Original transaction cannot be voided or captured | Send a void transaction for a declined transaction. |
| U13 | ORIG TRANS NOT FOUND | Transaction to be voided or captured was not found. | Send void transaction for the following trace number: `00000000-0000-0000-0000-000000000000` |
| U14 | BAD TYPE FOR ORIG TRANS | Void/Capture and original transaction types do not agree (CC/EFT) | Send a void credit card transaction for an echeck transaction. |
| U15 | ALREADY VOIDED ALREADY CAPTURED | Transaction was previously voided or captured | Void the same transaction twice. |
| U18 | UPDATE FAILED | Void or Capture failed | Send a transaction for $19.18 or $1918 |
| U19 | INVALID TRN | Account ABA number is invalid | Send echeck transaction with TRN of `123456789.` |
| U20 | INVALID CREDIT CARD NUMBER | Credit card number is invalid | Send a credit card transaction with a card number of `1111111111111111.` |
| U21 | BAD START DATE | Date is malformed | Send a transaction with scheduling data but a start date of `13/1/2008` or `1/1/2001.` |
| U22 | SWIPE DATA FAILURE | Swipe data is malformed | Send credit card transaction with `pg_cc_swipe_data=ABC123.` |

*Continued on next page*

 

# Understanding Response Parameters, Continued

, continued

| Code | Description | Comments | Test Parameters |
|---|---|---|---|
| U23 | INVALID EXPIRATION DATE | Malformed expiration date | Send credit card transaction with Ecom_Payment_Card_ExpDate_Month=13. |
| U25 | INVALID AMOUNT | Negative amount | Send a transaction for a negative amount (-$1.00). |
| U26 | INVALID DATA** | Invalid data present in the transaction | Send a void transaction with pg_original_authorization_code=. |
| U27 | CONV FEE NOT ALLOWED | Merchant configured for convenience fee but did not send one | For merchant configured to accept a convenience fee, send a transaction with an incorrect convenience fee in the pg_convenience_fee parameter. |
| U28 | CONV FEE INCORRECT | Merchant configured for convenience fee but did not send one | For merchants configured to accept a convenience fee, send a transaction with an incorrect convenience fee in the pg_convenience_fee parameter. |
| U29 | CONV FEE DECLINED | Convenience fee transaction failed – SplitCharge model only | N/A |
| U30 | PRINCIPAL DECLINED | Principle transaction failed – SplitCharge model only | N/A |
| U51 | MERCHANT STATUS | Merchant is not “live” | Send a transaction for a non-live Merchant ID. |
| U52 | TYPE NOT ALLOWED | Merchant not approved for transaction type (credit card or EFT) | Send a transaction of a type (credit card or echeck) that the account is not allowed to process. |
| U53 | PER TRANS LIMIT | Transaction amount exceeds merchant’s Per Transaction Limit (EFTs only) | Send a transaction that exceeds the merchant’s eCheck Limit(s). |
| U54 | INVALID MERCHANT CONFIG | Merchant’s configuration requires updating – call Customer Support | Send a transaction for $19.54 or $1954. |
| U80 | PREAUTH DECLINE | Transaction was declined due to preauthorization (Forte Verify) result | Send a transaction for $19.80 or $1980. |

*Continued on next page*

# Understanding Response Parameters, Continued

, continued

| Code | Description | Comments | Test Parameters |
|---|---|---|---|
| U81 | PREAUTH TIMEOUT | Preauthorizer not responding (Verify Only transactions only) | Send a transaction for $19.81 or $1981. |
| U82 | PREAUTH ERROR | Preauthorizer error (Verify Only transactions only) | Send a transaction for $19.82 or $1982. |
| U83 | AUTH DECLINE* | Transaction was declined due to authorizer declination | Send a transaction for $19.83 or $1983. |
| U84 | AUTH TIMEOUT | Authorizer not responding | Send a transaction for $19.84 or $1984. |
| U85 | AUTH ERROR | Authorizer error | Send a transaction for $19.85 or $1985. |
| U86 | AVS FAILURE AUTH | Authorizer's AVS Check failed | Send a transaction for $19.86 or $1986. |
| U87 | AUTH BUSY | Authorizing vendor busy; may be resubmitted (credit card only) | Send a transaction for $19.87 or $1987. |
| U88 | PREAUTH BUSY | Verification vendor busy; may be resubmitted (type 26 only) | Send a transaction for $19.88 or $1988. |
| U89 | AUTH UNAVAIL | Vendor service unavailable (credit card only) | Send a transaction for $19.89 or $1989. |
| U90 | PREAUTH UNAVAIL | Verification service unavailable (type 26 only) | Send a transaction for $19.90 or $1990. |
| U91 | Credit Card Not Allowed | Merchant account is configured to process only Debit Cards. Credit Cards not allowed | N/A |
| U92 | Debit Card Not Allowed | Merchant account is configured to process only Credit Cards. Debit cards not allowed | N/A |

*pg_response_description will contain the text of the vendor's response.
**pg_response_description will contain a more specific message.

*Continued on next page*

 

# Understanding Response Parameters, Continued

## Formatting Error Responses

The following table displays the codes returned when Forte finds formatting errors. The response description field will actually list all the offending fields in the message (to the 80-character limit). The description field will be formatted as follows:

```
<code>:<fieldname>[,<code>:<fieldname> ...]
```

The `pg_response_code` will contain the first error type encountered. All formatting errors begin with an `F`.

| Code | Description | Comments |
|---|---|---|
| F01 | MANDATORY FIELD MISSING | Required field is missing. |
| F03 | INVALID FIELD NAME | Name is not recognized. |
| F04 | INVALID FIELD VALUE | Value is not allowed. |
| F05 | DUPLICATE FIELD | Field is repeated in message. |
| F06 | CONFLICTING FIELD | Fields cannot both be present. |

## Fatal Exception Responses

The following table displays exceptions that will stop the processing of a well-formed message due to security or other considerations. All fatal exceptions begin with `E`.

| Code | Description | Comments |
|---|---|---|
| E10 | INVALID MERCH OR PASSWD | Merchant ID or processing password is incorrect. |
| E20 | MERCHANT TIMEOUT | Transaction message not received (I/O flush required?) |
| E55 | INVALID TOKEN | Specified token was invalid, could not be located, or may have been deleted. |
| | *Client Token Transactions* | For client token transactions where neither payment fields nor a payment token were specified, the client record does not have a Default Payment Method matching the transaction type. |
| | *Payment Token Transactions* | For payment token transaction where no client token is specified, the payment token must be clientless. |
| | *Both Client and Payment Tokens Present* | For transactions with client and payment tokens, the specified payment token is not associated with the client or is clientless. |
| E90 | BAD MERCH IP ADDR | Origination IP not on merchant's approved IP list. |
| E99 | INTERNAL ERROR | An unspecified error has occurred. |

# Integrating with SWP Checkout

## Overview

SWP Checkout supports two methods of integration: an HTML web form or transaction signing.

The HTML Integration method is the easiest way to integrate to the Forte platform. Merchants build a standard web page with a form that submits customer information to Forte. This integration method requires less work to implement and is ideal for charitable donations and simple, low-cost sales.

The transaction singing integration method requires a server-side technology to sign the message. The transaction needs to be hashed with an HMAC-MD5 signature using the secure transaction key set in Virtual Terminal.

## URLs

Use the following URLs when testing and submitting live transactions.

| Environment | URL |
|---|---|
| Sandbox | https://sandbox.paymentsgateway.net/swp/co/default.aspx |
| Live | https://swp.paymentsgateway.net/co/default.aspx |

## Using the HTML Integration Method

For a simple SWP Checkout integration, the HTML method is ideal for capturing customer data and submitting it to the Forte platform. When using this integration method, ensure that you set the **Accept Unsigned Transactions** field to **Yes** in Virtual Terminal.

Use the following sample code with this integration method:

```
<FORM METHOD="post" ACTION="https://sandbox.paymentsgateway.net/swp/co
/default.aspx">
<input name="pg_billto_postal_name_first" type="text" value="Bob"/>
<input name="pg_billto_postal_name_last" type="text" value="Smith"/>
<input type="hidden" name="pg_api_login_id" value="APILOGINID"/>
<input TYPE=SUBMIT>
</FORM>
```

*Continued on next page*

# Integrating with SWP Checkout, Continued

## Using the Transaction Signing Integration Method

The transaction signing method requires server-side technology to add a signature to the message. The signature hash is calculated with an HMAC-MD5 algorithm using keys created and maintained in Virtual Terminal (the **API Login ID** and the **Secure Transaction Key**).

Use the following hashing formula for your transaction signature:

```
pg_ts_hash = HMAC-MD5(pg_apilogin_id|pg_transaction_type|
pg_version_number|pg_total_amount|pg_utc_time|
pg_transaction_order_number,pg_secure_transaction_key)
```

To validate the hash in the postback, take the values of the `pg_utc_time` and the `pg_trace_number` parameters in the postback and insert them into the following formula:

```
pg_ts_hash_response = HMACMD5(pg_apilogin_id|pg_trace_number|
pg_total_amount|pg_utc_time,pg_secure_transaction_key)
```

The value of your validation hash should match the value of the postback parameter `pg_ts_hash_response`.

**NOTE:** Clients validating the response hash signature on merchant accounts set up with convenience fees will use the `pg_total_amount` value in the postback (which includes the convenience fee). For example, if the principal payment amount is $100 and the merchant account card fee is 2.5%, in the transaction signature the `pg_total_amount` is $100.00, but in the response hash validation the `pg_total_amount` will be $102.50.

Use the following sample code with this integration method:

```
<FORM METHOD="post" ACTION="https://sandbox.paymentsgateway.net/swp/co
/default.aspx">
<input name="pg_billto_postal_name_first" type="text" value="Bob"/>
<input name="pg_billto_postal_name_last" type="text" value="Smith"/>
<input type="hidden" name="pg_api_login_id" value="APILOGINID"/>
<input type="hidden" name="pg_transaction_type" value="10"/>
<input type="hidden" name="pg_version_number" value="1.0"/>
<input type="hidden" name="pg_total_amount" value="5.00"/>
<input type="hidden" name="pg_utc_time" value="634094514514687490"/>
<input type="hidden" name="pg_transaction_order_number"
value="100055"/>
<input type="hidden" name="pg_ts_hash" value="4bac0b9badbea7730cd41c33
4384bdfa"/>
<input TYPE=SUBMIT>
</FORM>
```

# Using Postback Information

## Overview

The following data can be provided in postbacks for both credit card and echeck transactions. As a best practice, Forte recommends sending an account number or other identifier (e.g., utility bill number, etc.) in either the `pg_consumerorderid` or `pg_wallet_id` fields.

## Sample Credit Card Transaction Postback

The following code sample displays an example credit card transaction postback:

```
pg_billto_postal_name_first=John
pg_billto_postal_name_last=Doe
pg_billto_postal_street_line1=500 W Bethany
pg_billto_postal_street_line2=Suite 200
pg_billto_postal_city=Allen
pg_billto_postal_stateprov=TX
pg_billto_postal_postalcode=75013
pg_billto_telecom_phone_number=866-290-5400
pg_billto_online_email=integration@forte.net
pg_consumerorderid=5
pg_wallet_id=5
pg_total_amount=0.01
pg_response_description=APPROVED
pg_response_code=A01
pg_response_type=A
pg_trace_number=c07d219e-edba-40f3-ada7-c66ef25d10b9
pg_transaction_type=10
pg_authorization_code=809965
pg_last4=7062
pg_payment_card_type=visa
pg_payment_card_expdate_month=07
pg_payment_card_expdate_year=2016
```

*Continued on next page*

# Using Postback Information, Continued

**Sample eCheck Transaction Postback**

The following code sample displays an example echeck transaction postback:

```
pg_billto_postal_name_first=Jane
pg_billto_postal_name_last=Doe
pg_billto_postal_street_line1=500 W Bethany
pg_billto_postal_street_line2=Suite 200
pg_billto_postal_city=Allen
pg_billto_postal_stateprov=TX
pg_billto_postal_postalcode=75013
pg_billto_telecom_phone_number=866-290-5400
pg_billto_online_email=integration@forte.net
pg_consumerorderid=3
pg_wallet_id=4
pg_total_amount=1.00
pg_response_description=APPROVED
pg_response_code=A01
pg_response_type=A
pg_trace_number=cc351c20-3c35-4adc-8652-fe9d684b3485
pg_transaction_type=20
pg_authorization_code=21042701
pg_last4=3333
```