Community Gift Cards

Webhooks

What are Webhooks ?

When CommunityGiftCards customers (gift card holders) want to use their gift cards at your place of business, by redeeming a portion (or all) of their gift card in exchange for goods or services that you sell, you or your employee will be scanning a QR code (or clicking on a link that the customer received via e-mail).

When you do, CommunityGiftCards will show you the current balance on the card, and will provide you with the option to **Redeem an Amount from the Card**.

There are a few different options for integration between CommunityGiftCards and your own system.

No Integration – Notification only: Using this option, the CommunityGiftCards system can be configured to simply send you an e-mail after you've specified the amount. The e-mail will contain the amount redeemed.

Stored Items: You can choose to store a copy of your items within the CommunityGiftCards system, such as what goods or services you have for sale, their price, and the maximum quantity that can be selected at redemption time. With this method, a notification e-mail can be sent to you, containing the list of items that were redeemed, how many of each, the price of the item, and the amount redeemed from the card.

Point of Sale (POS) systems: You may already have your own Point-of-Sale system, that you want CommunityGiftCards to communicate with. Using this method, CommunityGiftCards can "pull" available items from your POS system, and after the selections are made, can also "push" updates to your POS system so that the inventory can be updated, payments via gift cards are captured, etc.

Webhooks: If you have your own custom tool/system that can undergo some development effort, you may have the option of creating a custom integration between that system and CommunityGiftCards. This document describes how webhooks are used, and what work your developer needs to do in order to integrate your system to CommunityGiftCards.

Webhooks are a way to implement the custom integration that is needed in order to minimize effort related to reconciliation, item selection, payment capture, and inventory updates.

With this method, the Webhook specifies at redemption time what items are available, and after the redemption is complete, your system is notified with details of what items were selected, how many, and the redemption amount. An example where a Webhook could be used, is for a Property Management System (PMS). When it is time to redeem a gift card at a hotel, the hotel's PMS and CommunityGiftCards will communicate, so that your front desk personnel can simply choose the guest's name and have the payment reflected in the PMS, updating the balanced owed.

NEW "Call to Action" functionality added April 2018: In addition to webhooks being used for card redemption activities, webhook functionality has been augmented to allow a "call to action" when card holders are looking at where their gift card can be used, in the Marketplace. For example, if a gift card holder is in the Marketplace, and the locations being displayed are accommodations providers, a "call to action" can be something that allows the gift card holder to "Check Availability".

There are two parts to the call to action functionality.

The first part consists of the CommunityGiftCards platform prompting via the webhook function, whether a call to action is available. Webhook developers will need to respond to the "mode" parameter being set to "**calltoactioninfo**". In this instance, there will be no location_code specified in the URL, and your system should respond with specifics regarding what the call to action is, what fields need to be asked of the gift card holder, etc.

The second part consists of the CGC platform interacting with the card holder, using the information provided by your webhook (such as what fields need to be provided by the gift card holder, etc.). After the gift card holder has specified the information required, the CGC platform will communicate via the webhook with your system, specifying the mode "**calltoactionexecute**".

Please read the section titled "Call to Action Specifications" for complete details.

How do Webhooks work ?

In order for two computerized systems to communicate with each other, a ***protocol*** must first be defined. This protocol specifies the formats of the messages that can be passed back and forth, the rules that are implemented when there is missing information in those messages, and the results that are expected when messages are received by each party in the communication.

CommunityGiftCards has defined a protocol, and a Webhook is simply a URL that can be used to "access" your system communication point.

The system administrator can create Webhooks as necessary, and can create a webhook for your place of business...you just need to tell the administrator what the URL is for your system.

There are two types of communications that occur using this Webhook:

- Item selection
- Assignment of card redemption

Overall workflow

Step 1: A customer wants to use their gift card, to purchase goods/services at your place of business, and notifies your employee.

Step 2: Your employee uses a device that is already known to CommunityGiftCards, to either scan a QR code that your customer provides him/her (on their phone or on a piece of paper which is a print-out of the e-mail they received when the gift card was bought); OR your employee asks your customer to forward the e-mail, and the link in that e-mail is clicked on, using the device.

Step 3: A page on CommunityGiftCards is displayed, that shows the current balance on the card. Your employee selects the "**Redeem an Amount from the Card**" option.

Step 4: (webhook communication) The CommunityGiftCards system, communicates with your system using the Webhook, issuing an "Item selection" request. Your system responds with information about which items are available for selection, their prices, and descriptions.

Step 5: Your employee selects items (potentially also including multiple quantities of those items) that your customer wants to purchase. The CommunityGiftCards system updates the screen as selections are made or undone, showing what the remaining balance on the gift card would be, how much "extra" the customer would have to pay (if their gift card balance isn't enough to cover all the items selected), and highlights what the redemption amount from the card is. Once done, your employee selects the "**Redeem Card**" option.

Step 6: (webhook communication) The CommunityGiftCards system updates the balance on the card, and sends a communication to your system via the Webhook, regarding which items were selected, quantity, price, and total gift card redemption amount.

Step 7: Your system uses the information passed to it via the Webhook, to update its internal settings, quantities on hand, etc.

Technical Specifications

In this section, the technical implementation details of the Webhook and the protocol are defined.

This is the section that your system developer needs to pay close attention to, in order to ensure that your system is properly integrated with CommunityGiftCards.

As stated above, there are two types of communications that need to be implemented:

- Item selection
- Assignment of card redemption

For the purposes of this section, your Webhook URL will be referred to as: WEBHOOK_URL … typically, it is a URL that is of the following format: http://www.yourdomain.com/yourwebhook. This URL should point to a "listener" on your system that waits for messages to be sent to it, by CommunityGiftCards.

Item selection

Using your WEBHOOK_URL as the base, CommunityGiftCards will add a few parameters to the end of the URL as follows:

- **mode**: will be "select"
- **location_code**: this is the unique location code for the location that is doing the redemption. If your WEBHOOK_URL is only used by your place of business, then this can safely be ignored. If, on the other hand, the WEBHOOK_URL is being used by multiple locations, the location_code can be used to determine which specific location the item list needs to be related to.
- **cardbalance**: This is the balance currently remaining on the card. You can use this to filter your item list if you wish.

Note: If your WEBHOOK_URL already has some parameters included, CommunityGiftCards will simply append the parameters above to the end.

Responses from your webhook should be formatted as valid XML … here is the response format expected:

```
<CGCResponse>
        <SelectionIdTitle>My contract ID</SelectionIdTitle>
        <MaxRedeemTitle>Price</MaxRedeemTitle>
        <DescriptionTitle>My item description</DescriptionTitle>
        <ShowSelectionId>TRUE</ShowSelectionId>
        <ShowMaxRedeem>TRUE</ShowMaxRedeem>
        <SelectionList>
                <Selection>
                        <SelectionId>uniqueid1</SelectionId>
                        <MaxRedeem>123.45</MaxRedeem>
                        <Description>This is item 1</Description>
                        <MaxQuantity>10</MaxQuantity>
                </Selection>
                <Selection>
                        <SelectionId>uniqueid2</SelectionId>
                        <MaxRedeem>44.99</MaxRedeem>
                        <Description>This is item 2</Description>
                        <MaxQuantity>3</MaxQuantity>
                </Selection>
        </SelectionList>
</CGCResponse>
```

SelectionIdTitle: When CommunityGiftCards displays the selectable items, the unique identifier of each item is displayed under a column heading … this field specifies what that column heading should be. The default value is “Unique Id” if you do not provide it in the XML. Note that you can choose not to display the unique identifier heading at all (see below).

MaxRedeemTitle: The maximum redeem amount/price for each item is displayed under a column heading specified by the value for this field. The default value is “Maximum amount” if you do not provide it in the XML. Note that you can choose not to display this heading at all (see below).

DescriptionTitle: The description of each item is always displayed. The value in this field specifies what the column heading should be. The default value is “Description” if you do not provide it in the XML.

ShowSelectionId: This field specifies whether the unique identifier of each item, is actually displayed or not. The default value is “TRUE” if you do not provide it in the XML.

ShowMaxRedeem: This field specifies whether the price/maximum redeem amount for each item, is actually displayed or not. The default value is “TRUE” if you do not provide it in the XML.

SelectionList: an XML collection that contains 1 or more “Selection” collections

Selection: an XML collection that provides details for each item that is selectable by your employee.

The following 4 fields define the details for each item:

SelectionId: the unique identifier for the item (this will be used later during “assignment” to provide your webhook with the details of which items were selected.

MaxRedeem: the price/max redeem amount for the item.

Description: the item's description.

MaxQuantity: how many maximum quantity that can be selected by your employee. Note that if you do not provide this field in the XML, the default value is 1. If you provide a value larger than 10, the system will only display up to 10 in the selectable quantity field.

(optional) PreTaxRedeem: gift card holders earn points on their cards (which, when vested 24 hours later, turn into true “value”), depending on whether the location doing the redemption has opted in to doing so. You wouldn't want the gift card holder to earn a percentage based on the total price, so this field can be provided to provide the pre-tax price, on which the rewards system is based. If you do **not** provide this field, the user will be forced to provide the pre-tax total on the redemption screen.

Assignment of card redemption

Using your WEBHOOK_URL as the base, CommunityGiftCards will add a few parameters to the end of the URL as follows:

- **mode**: will be "assign"
- **location_code**: this is the unique location code for the location that is doing the redemption. If your WEBHOOK_URL is only used by your place of business, then this can safely be ignored. If, on the other hand, the WEBHOOK_URL is being used by multiple locations, the location_code can be used to determine which specific location the redemption is occurring for.
- **card_code**: The CommunityGiftCards unique card code that was redeemed. You can choose to ignore this, or, for reference purposes, your system can store its value in its tables. NOTE that what will be provided here is an "obfuscated" version of the card code.
- **numitems**: This parameter will indicate how many items were selected.
- **itemid[x]**: Many of these could be provided, and they indicate the unique identifiers that were included in the item selection. For example, if numitems is 2, there will be 2 parameters in the URL, such as: itemid1=abc and itemid2=def
- **itemqty[x]**: Specifies how many of each item was selected by your employee.
- **itemamount[x]**: Specifies how much of the card redemption, can be allocated to each of the items selected. Please see the section below entitled "**Calculating Item Redemption Amounts**", for details on how CommunityGiftCards determines this amount.

Note: If your WEBHOOK_URL already has some parameters included, CommunityGiftCards will simply append the parameters above to the end.

Responses from your webhook should be formatted as valid XML … here is the response format expected:

```
<CGCResponse>
        <Status>OK</Status>
        <ErrorMessage></ErrorMessage>
        <UniqueTransactionId>12345</UniqueTransactionId>
        <UniqueTransactionId>67890</UniqueTransactionId>
        <UniqueTransactionId>22222</UniqueTransactionId>
</CGCResponse>
```

Status: should either be "OK" or "ERROR". OK means that your system was able to successfully process the assignment. ERROR means that there was a problem with your system being able to process the assignment.

Note that if the webhook returns ERROR as the Status value, then the redemption within CommunityGiftCards will NOT take place, and an error message will be displayed to your employee.

ErrorMessage: if the Status value is "ERROR", you should also provide an ErrorMessage value. This message will be displayed to your employee. If the Status value is "ERROR", but you do not provide an ErrorMessage value, the message that will be displayed to your employee is "Unknown error occurred from webhook".

UniqueTransactionId: you can have 1 or more of these provided in the XML (or none if your system does not generate unique transaction identifiers for every assignment that occurs within it). These values will be stored for reference purposes only. Note that these are ignored entirely if the Status value is "ERROR".

Calculating Item Redemption Amounts

When the CommunityGiftCards system communicates with your system using your WEBHOOK_URL, it provides information pertaining to which items were selected by your employee, how many of each item, and the applicable redemption amount that should be allocated to each item.

The last bit of information (applicable redemption amount for each item), is calculated based on a few different factors, and this section will provide further clarity on those factors, and will also illustrate by way of pseudo-code, what the resulting amounts would be.

- Factor 1 – item list order: during the Item Selection part of the process, CommunityGiftCards communicates with your system via your webhook, and that webhook returned a number of items that were selectable at that time. It is important to note that CommunityGiftCards will display those items to your employee, in the order in which they were defined in the XML by your webhook.
- Factor 2 – card balance: the current balance on the card is used as the basis for what can be redeemed from the card, and that balance is the maximum amount that will be "allocated" to the items selected by your employee.
- Factor 3 – item selections: which items are selected by your employee (and potentially how many of each item if the MaxQuantity field was provided during the Item Selection retrieval), also help define the per-item redeem amount.

It should be noted that CommunityGiftCards will NOT limit your employee to selecting only the amount of items that can be redeemed by the gift card … they can select additional items, and the system will display the amount that is "over-allocated" in very visible font (bright red background, and bold black characters).

Every time a selection is updated by your employee, the system will recalculate all of the amounts. Updates include selecting a non-zero quantity from the drop-down list (in the cases where MaxQuantity was provided and is not 1), or putting a check-mark next to an item (where the MaxQuantity is not provided or is 1).

Here's some example pseudo-code that describes the approach taken when doing the recalculation. In the example below, the "**item redeem value**" is what will be sent as the "itemamount[x]" parameter.

```
Set "total value selected" to zero.
Set "card balance" to the balance on the gift card in question.

Go through every item displayed on screen (all items returned from the "Item Listing" part of the process),
in the order in which they were received
          Set "item price" to be the price of the item (MaxRedeem from the Item Listing XML)

          Set "quantity selected" either to 1 or 0 (if MaxQuantity is 1), or to the quantity selected by the
          employee (if MaxQuantity was greater than 1)

          If the "quantity selected" is non-zero:
                    Set "item value" to be the "quantity selected" times the "item price"

                    If "item value", plus "total value selected" is greater than "card balance":
                              Set "item value left" to "card balance" minus "total value selected"
                              If "Item value left" is <= zero, then the "item redeem value" is zero
                              Else "item redeem value" is set to "item value left"

                    Else
                              Set "item redeem value" to "item value"

                    Increment "total value selected" by "item value"

The "total redeem amount" for the card, will be "total value selected" or "card balance", whichever is
smaller.

The screen will also display what amount is above and beyond what the card can be redeemed for, if the "total
value selected" is greater than the "card balance".

NOTE:  A very similar approach is taken for the "rewards based on" calculation and display,
which the user can override.
```

Call to Action Specifications

As stated previously, a call to action is a feature by which the CommunityGiftCards platform can interact with the gift card holder via the Marketplace (displayed when card holders want to see where they can use their card).

If your system has no such call to action, then your Webhook could simply "ignore" the modes "calltoactioninfo" and "calltoactionexecute" … more specifically, your system would likely respond with an error of some kind.

The CommunityGiftCards platform will only display a call to action when your system has successfully responded with information that details what the call to action is.

Part 1: Call to Action Info (mode = "calltoactioninfo")

In a similar fashion to other webhook functionality, the CGC platform will use your WEBHOOK_URL as the base, and will add one parameter to the end of the URL as follows:

- **mode**: will be "calltoactioninfo"

Note that the CGC platform does **not** provide the location_code in this instance. Your system should respond with information regarding which location codes the call to action is applicable to. It is done in this way so as to ensure that the length of the URL is within allowable limits (adding multiple location codes to the URL might potentially cause errors).

Your webhook should respond with properly formatted XML, similar to other webhook functionality, wrapped with a "CGCResponse" tag … the table below describes every tag and the expectations for formatting, values, etc.

```
<CGCResponse>
        <ButtonLabel>Check Availability</ButtonLabel>
        <GoLinkLabel>Book Now</GoLinkLabel>
        … other tags ...
</CGCResponse>
```

Tag	Description
LocationCode1 … LocationCode[x]	Required. Indicates the location codes that "qualify" for the call to action.
ButtonLabel	Required. The label to display on the button that the gift card holder can click on in order to perform the call to action. Example: "Check Availability"
GoLinkLabel	Required. **AFTER** the CGC platform has executed the call to action (see Part 2), some locations will be the potential "targets" for the action. For those targets, the CGC platform will display a link or button which will allow the card holder to proceed. This is the label for that link/button. Example: "Book Now".
GoLinkBuild	Optional (default = FALSE). Setting this to "TRUE" indicates that the CGC platform should use the URL as stored in the CGC platform for the location, as the base URL that the gift card holder will access.
GoLinkIndicatorField	Optional (default = "fromcgcwebhook"). The CGC platform will always add a parameter to the URL, indicating that the gift card holder is accessing the URL from within the CGC platform – by default, the parameter name will be "fromcgcwebhook" and the value will be "1". Specifying this field allows you to customize what that parameter name will be.
GoLinkIndicatorValue	Optional (default = "1"). As per the above, an additional parameter is always added to the URL that will be accessed by the gift card holder. Specifying this field allows you to customize the parameter value that will be included.
GoLinkYesNoValueTrue / GoLinkYesNoValueFalse	Optional (default = "1" for GoLinkYesNoValueTrue and "0" for GoLinkYesNoValueFalse). When GoLinkBuild is "TRUE", for any YesNo fields, CGC will normally include a value of either "1" (for Yes), or "0" (for No) for the parameter in the link.
GoLinkCheckboxOnValue	Optional (default = "1"). When GoLinkBuild is "TRUE", for any Checkbox fields, CGC will normally include a value of "1" when the gift card holder has placed a check-mark.
GoLinkDateFormat	Optional (default = "YYYYMMDD"). When GoLinkBuild is "TRUE", for any Date fields, CGC will normally include the date selected by the gift card holder, with YYYYMMDD format. Specifying this tag enables the system to send the date in a different format. Valid values are YYYYMMDD, DD-MON-YYYY, YYYY-MM-DD.
FieldName1 … FieldName[x]	Required. The internal field name that will be used to pass back information to your webhook in order to execute the call to action. Examples: ArrivalDate, NumNights, NumGuests, etc.
FieldGoLinkParmName1 … FieldGoLinkParmName[x]	Conditional. If GoLinkBuild is set to "TRUE", then this tag is required. This indicates the parameter name that gets added to the URL for this field.
FieldLabel1 … FieldLabel[x]	Required. The label that is used to display to the card holder. Examples: "Arrival Date", "# of Nights", "# of Guests", etc.
FieldFormat1 … FieldFormat[x]	Required. Indicates the type of input that the gift card holder needs to provide. Valid values are "Number", "Date", "YesNo", or "Checkbox". Depending on this value, other tags may be required (next few rows).

If format is “Number”:	FieldMinimum[x] and FieldMaximum[x] are **required** and must be integers. FieldDefaultValue[x] – optional (default is FieldMinimum[x]) – by default, when the call to action is first displayed, what is the default pre-selected value in the drop-down list. GoLinkDoDateTranslation[x] – optional (default FALSE). TRUE indicates that if GoLinkBuild is also TRUE, then what will be added to the URL will be a date (formatted according to GoLinkDateFormat) instead of a number. If this tag is set to TRUE, then the next tag is required. GoLinkDateTranslationBasedOnFieldName[x] – conditional (**required** if GoLinkDoDateTranslation[x] is set to TRUE). Indicates the field name (FieldName[x]) on which the date translation is based. GoLinkDateTranslationAddOneDay[x] – optional (default FALSE). If date translation occurs, the system will by default make the dates inclusive. As an example, if the number selected by the gift card holder is “1”, then the date included in the URL that is built, will be the same value as the value for the field name identified by GoLinkDateTranslationBasedOnFieldName[x]. If instead you want the date to be included in the URL to be the next day, then set this tag to TRUE.
If format is “Date”	FieldCanBePast[x] – optional (default FALSE). Indicates whether the date chosen by the gift card holder can be in the past or not. Note that the format of what will be sent by the CGC platform to your webhook, for Date fields, will be YYYYMMDD, but the format of dates included in the URL that is built, is driven by GoLinkDateFormat.
If format is “YesNo”	The CGC platform will present a drop-down list to the gift card holder. The drop-down list will contain a “-- Please select --” option, and until the gift card holder selects either the “Yes” or “No” option, the CGC platform will *not* execute the call to action and will prompt the gift card holder to select an option. For this type of field, the GoLinkYesNoValueTrue tag determines what value that CGC will include in the URL, but the values “yes” or “no” will be sent to your webhook.
If format is “Checkbox”	The CGC platform will display a check-box for the field, and the gift card holder will either put a check-mark in the box or not. For this type of field, the GoLinkCheckboxOnValue tag determines what value that CGC will include in the URL, but the values “on” or “off” will be sent to your webhook.

Part 2: Call to Action Execute (mode = “calltoactionexecute”)

In similar fashion to other webhook functionality, the CGC platform will use your WEBHOOK_URL as the base, and will add other parameters to the end of the URL as follows:

- **mode**: will be “calltoactionexecute”
- others based on the information provided by your webhook in part 1 … the parameter name will be governed by the FieldName[x] tag, and the parameter value will be governed by the FieldFormat[x] tag (“Number” means the parameter value will be an integer, “Date” means the parameter value will be a date in the “YYYYMMDD” format, “YesNo” means the parameter value will be either “yes” or “no”, and “Checkbox” means the parameter value will be either “on” or “off”).

Note that the CGC platform does **not** provide the location_code in this instance. Your system should respond with information regarding which location codes qualify, as a result of the call to action being executed.

The response from your webhook is fairly simple (again, wrap it with “CGCResponse”).

```
<CGCResponse>
        <LocationCode1>abc</LocationCode1>
        <URL1>http://www.yourdomain.com/whatever</URL1>
        … other tags ...
</CGCResponse>
```

Tag	**Description**
LocationCode1 … LocationCode[x]	Required. Indicates the location codes that “qualify” as a result of the call to action execution.
URL1 … URL[x]	Conditional. It the webhook response from part 1, had a value of “TRUE” for GoLinkBuild, then you do **not** need to provide these tags, as the CGC platform will determine the URL that the gift card holder needs to access based on the tags specified in part 1. However, if GoLinkBuild is either not present or is set to “FALSE”, then this tag is required for every location.