ccrz.cc_hk_Subscriptions

CloudCraze has added a rich set of data structures and methods for allowing customers to purchase items on a recurring basis. This extension point provides a way for implementors to extend that functionality.

- global ccrz.cc_hk_Subscriptions getInstance(Map<String,Object> inputData)
- global virtual Map<String,Object> fetchSubscriptionOptions(Map<String,Object> params)
- global virtual Map<String,Object> fetchSubProdTerms(Map<String,Object> params)
- global virtual Map<String,Object> filterSubscriptionOptions(Map<String,Object> params)
- global static Map<String,Object> convertSubProdTermToMap(ccrz__E_SubProdTerm__c spt)
- global static List<Map<String, Object>> convertModifierSubProdTermsToMap(List<ccrz__E_SubProdTerm__c> spts)
- global static Map<String,Object> convertSubscriptionToMap(ccrz__E_Subscription__c theSubscription)
- global virtual Map<String,Object> fetchSubscriptions(Map<String,Object> inputData)
- global virtual Map<String,Object> cancelSubscription(Map<String,Object> inputData)
- global virtual Map<String,Object> fetchSubscription(Map<String,Object> inputData)
- global virtual Map<String,Object> updateSubscriptionPayment(Map<String,Object> inputData)
- global virtual Map<String,Object> onOrder(Map<String,Object> inputData)
- global virtual Map<String,Object> calculateTargetDate(final Map<String,Object> inputData)
- global virtual Map<String,Object> createSubscription(Map<String,Object> inputData)
- global virtual Map<String,Object> getSubscriptionProcessQuery(Map<String,Object> inputData)
- global virtual Map<String,Object> preProcessSubscriptions(Map<String,Object> inputData)
- global virtual Map<String,Object> processSubscriptions(Map<String,Object> inputData)
- global virtual Map<String,Object> postProcessSubscriptions(Map<String,Object> inputData)
- global virtual Map<String,Object> processRecurrOrders(Map<String,Object> inputData)
- global virtual Map<String,Object> parsePaymentData(Map<String,Object> inputData)
- global virtual Map<String,Object> createTransactionPayment(Map<String,Object> inputData)
- global virtual Map<String,Object> processRecurrInstallments(Map<String,Object> inputData)
- global virtual Map<String,Object> processRecurrCombined(Map<String,Object> inputData)
- global virtual Map<String,Object> processRecurrErrors(Map<String,Object> inputData)
- global virtual Map<String,Object> recurrOrderAddrInfo(Map<String,Object> inputData)
- global virtual Map<String,Object> recurrOrder(Map<String,Object> inputData)
- global virtual Map<String,Object> getSubscriptionRenewalQuery(Map<String,Object> inputData)
- global virtual Map<String,Object> renewAddrInfo(Map<String,Object> inputData)
- global virtual Map<String,Object> renewSubscription(final Map<String,Object> inputData)
- global virtual Map<String,Object> processRenewals(Map<String,Object> inputData)

global ccrz.cc_hk_Subscriptions getInstance(Map<String,Object> inputData)

Retrieves the currently configured ccrz.cc_hk_Subscriptions implementation. If no configuration is found this will return a new instance of ccrz.cc_hk_Subscriptions. The storefront may also be specified as an input parameter. If no storefront is passed then the current context storefront is used.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront to use to determine which ccrz.cc_hk_Subscriptions implementation to instantiate. If no storefront is passed the current context storefront is use

global virtual Map<String,Object> fetchSubscriptionOptions(Map<String,Object> params)

Returns the list of subscription options available for a given set of products. This method is called for the product detail and product list pages as well as from the featured products component on the home page and where ever related products are displayed. To summarize, anywhere that a product is being displayed for merchandising purposes, as opposed to checkout or order history, this method will be called.

The OOTB implementation returns the mapped ccrz__E_SubProdTerm__c records associated with each product filtered by the ccrz__Enabled__c, ccrz__EffectiveStart__c, and ccrz__EffectiveEnd__c fields and ordered by ccrz__Sequence__c.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_NAME_PROD_ID

The List<String> of product ids for which to return the subscription data.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_NAME_SUBPRODTERMS

A Map<String,List<Map<String,Object>>> keyed by the product id requested which then maps to a list of the converted ccrz__E_SubProdTerm__c data for the given subscription option. See convertSubProdTermToMap.

global virtual Map<String,Object> fetchSubProdTerms(Map<String,Object> params)

Given a set of ids for ccrz__E_SubProdTerm__c this method returns the mapped record data including child modifiers (child subscription data). See convertSubProdTermToMap. This method is used when displaying the cart and historical order data.

The OOTB implementation takes the list of ids, filters by the ccrz__Enabled__c, ccrz__EffectiveStart__c, and ccrz__EffectiveEnd__c fields and returns the list ordered by ccrz__Sequence__c.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_NAME_SUBPRODTERMS

The List<String> of ccrz__E_SubProdTerm__c ids to retrieve.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_NAME_SUBPRODTERMS

The Map<String,Map<String,Object>> of retrieved data keyed by id.

global virtual Map<String,Object> filterSubscriptionOptions(Map<String,Object> params)

This method is called after fetchSubscriptionOptions after the subscriptions have been priced and is meant to provide a means to simply filter which subscription options are returned without having to perform the fetch as well.

The OOTB implementation removes any subscriptions options that do not have a price. It makes allowances for the "No Subscription Option" which may not yet have a price and does not remove that if it is in the list. It also by default sets the first item in the list as "checked" for display purposes.

Inputs

A Map<String,Map<String,Object>> of subscription data mapped by id. See convertSubProdTermToMap for the mapping detail.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_NAME_SUBPRODTERMS

The List<Map<String,Object>> of filtered mapped subscription options.

global static Map<String,Object> convertSubProdTermToMap(ccrz__E_SubProdTerm__c spt)

convertSubProdTermToMap is a utility method that converts a ccrz__E_SubProdTerm__c record to a general Map<String,Object> for presentation on the user interface. The return of the method is defined as:

```
return new Map<String,Object>{
    'sfid' => spt.Id
    , 'name' => spt.Name__c
    , 'pdpDisplayName' => pdpDisplayNameTranslated
    , 'pdlDisplayName' => (String.isBlank(pdlDisplayNameTranslated))?
pdpDisplayNameTranslated:pdlDisplayNameTranslated
    , 'orderDisplayName' => (String.isBlank
(orderDisplayNameTranslated))?pdpDisplayNameTranslated:
orderDisplayNameTranslated
    , 'cartDisplayName' => (String.isBlank(cartDisplayNameTranslated))?
pdpDisplayNameTranslated:cartDisplayNameTranslated
    , 'emailDisplayName' => (String.isBlank
(emailDisplayNameTranslated))?pdpDisplayNameTranslated:
emailDisplayNameTranslated
    , 'displayName' => displayNameTranslated
    , 'subProdTermId' => spt.SubProdTermId__c
    , 'storefront' => spt.Storefront__c
    , 'sequence' => spt.Sequence__c
    , 'orderFrequency' => spt.OrderFrequency__c
    , 'orderFrequencyUOM' => spt.OrderFrequencyUOM__c
    , 'orderCount' => spt.OrderCount__c
    , 'installmentFrequency' => spt.InstallmentFrequency__c
    , 'installmentFrequencyUOM' => spt.InstallmentFrequencyUOM__c
    , 'installmentCount' => spt.InstallmentCount__c
    , 'ignoreInstallmentFrequencyFields' => spt.
IgnoreInstallmentFrequencyFields__c
    , 'enabled' => spt.Enabled__c
    , 'effectiveStart' => spt.EffectiveStart__c
    , 'effectiveEnd' => spt.EffectiveEnd__c
    , 'parentSubscription' => spt.CCSubProdTerm__c
    , 'autoRenew' => spt.AutoRenew__c
    , 'product' => spt.CCProduct__c
    , 'modifierSubscriptions' => (spt.CCSubProdTerms__r.size() > 0)?
convertModifierSubProdTermsToMap(spt.CCSubProdTerms__r):null
};
```

Note that the values for pdlDisplayNameTranslated, etc. are the mapped localized names from either the main ccrz__E_SubProdTerm__c record or the corresponding ccrz__E_SubProdTermI18N__c record as defined by normal CloudCraze localization fallback rules.

global static List<Map<String, Object>> convertModifierSubProdTermsToMap(List<ccrz__E_SubProdTerm__c> spts)

convertModifierSubProdTermsToMap converts a list of child ccrz__E_SubProdTerm__c object to a general data structure to be used in the user interface. The implementation is defined as:

```
List<Map<String, Object>> sptModifiers = new List<Map<String,
Object>>();
for(E_SubProdTerm__c spt : spts){
    sptModifiers.add(
        new Map<String,Object>{
            'sfid' => spt.Id
            , 'orderCount' => spt.OrderCount__c
            , 'installmentCount' => spt.InstallmentCount__c
            , 'ignoreInstallmentFrequencyFields' => spt.
IgnoreInstallmentFrequencyFields__c
            , 'autoRenew' => spt.AutoRenew__c
            , 'sequence' => spt.Sequence__c
        }
    );
}

return sptModifiers;
```

global static Map<String,Object> convertSubscriptionToMap(ccrz__E_Subscription__c theSubscription)

This is a utility method to convert a ccrz__E_Subscription__c object to a generalized data structure for display in the user interface. The method is defined as:

```
return new Map<String,Object>{
     'sfid' => theSubscription.Id
    ,'name' => theSubscription.Name
    ,'account' => (null==theSubscription.Account__c)?null:new
Map<String,Object>{'sfid' => theSubscription.Account__c, 'name' =>
theSubscription.Account__r.Name, 'accountNumber' => theSubscription.
Account__r.AccountNumber, 'accountGroupSfid' => theSubscription.
Account__r.E_AccountGroup__c}
    ,'autoRenew' => theSubscription.AutoRenew__c
    ,'currencyIsoCode' => theSubscription.CurrencyISOCode__c
    ,'ignoreInstallFreqFields' => theSubscription.
IgnoreInstallmentFrequencyFields__c
    ,'instAddr' => (null==theSubscription.InstallmentAddress__c)?null:
convertContactAddressToMap(theSubscription.InstallmentAddress__r)
    ,'instCountRemaining' => theSubscription.
InstallmentCountRemaining__c
    ,'instLastDate' => (null==theSubscription.
InstallmentLastDate__c)?'':theSubscription.InstallmentLastDate__c.
date().format()
    ,'instNextDate' => (null==theSubscription.
InstallmentNextDate__c)?'':theSubscription.InstallmentNextDate__c.
date().format()
```

```
    ,'instStatus' => theSubscription.InstallmentStatus__c
    ,'orderAddr' => (null==theSubscription.OrderAddress__c)?null:
convertContactAddressToMap(theSubscription.OrderAddress__r)
    ,'orderCountRemaining' => theSubscription.OrderCountRemaining__c
    ,'orderLastDate' => (null==theSubscription.OrderLastDate__c)?'':
theSubscription.OrderLastDate__c.date().format()
    ,'orderNextDate' => (null==theSubscription.OrderNextDate__c)?'':
theSubscription.OrderNextDate__c.date().format()
    ,'orderStatus' => theSubscription.OrderStatus__c
    ,'parentSub' => (null==theSubscription.ParentSubscription__c)?null:
new Map<String,Object>{'sfid' => theSubscription.ParentSubscription__c,
'name' => theSubscription.ParentSubscription__r.Name}
    ,'quantity' => theSubscription.Quantity__c
    ,'recurringPrice' => theSubscription.RecurringPrice__c
    ,'recurringPriceSubAmt' => theSubscription.RecurringPriceSubAmt__c
    ,'sequence' => theSubscription.Sequence__c
        ,'spt' => theSubscription.SPT__c
        ,'sptInstCount' => theSubscription.SPTInstallmentCount__c
        ,'sptInstFreq' => theSubscription.
IgnoreInstallmentFrequencyFields__c ? theSubscription.
SPTOrderFrequency__c : theSubscription.SPTInstallmentFrequency__c
        ,'sptInstFreqUOM' => theSubscription.
IgnoreInstallmentFrequencyFields__c ? theSubscription.
SPTOrderFrequencyUOM__c : theSubscription.SPTInstallmentFrequencyUOM__c
        ,'sptOrderCount' => theSubscription.SPTOrderCount__c
        ,'sptOrderFreq' => theSubscription.SPTOrderFrequency__c
        ,'sptOrderFreqUOM' => theSubscription.SPTOrderFrequencyUOM__c
        ,'storedPayment' => (null==theSubscription.StoredPayment__c)?
null:cc_hk_Payment.convertStoredPaymentToMap(theSubscription.
StoredPayment__r)
        ,'storefront' => theSubscription.Storefront__c
        ,'subscriptionId' => theSubscription.SubscriptionId__c
        ,'subProduct' => theSubscription.SubscriptionProduct__c
        ,'subProductSku' => theSubscription.SubscriptionProduct__r.
SKU__c
        ,'subStartDate' => (null==theSubscription.
SubscriptionStartDate__c)?'':theSubscription.SubscriptionStartDate__c.
format()
        ,'subStatus' => theSubscription.SubscriptionStatus__c
};
```

and convertContactAddressToMap is defined as:

```
return new Map<String,Object>{
     'sfid' => theAddress.Id
    ,'name' => theAddress.Name
    ,'address1' => theAddress.AddressFirstline__c
    ,'address2' => theAddress.AddressSecondline__c
    ,'address3' => theAddress.AddressThirdline__c
    ,'city' => theAddress.City__c
    ,'companyName' => theAddress.CompanyName__c
    ,'daytimePhone' => theAddress.DaytimePhone__c
    ,'email' => theAddress.Email__c
    ,'firstName' => theAddress.FirstName__c
    ,'homePhone' => theAddress.HomePhone__c
    ,'lastName' => theAddress.LastName__c
    ,'mailStop' => theAddress.MailStop__c
    ,'middleName' => theAddress.MiddleName__c
    ,'partnerId' => theAddress.Partner_Id__c
    ,'postalCode' => theAddress.PostalCode__c
    ,'state' => theAddress.State__c
    ,'stateCode' => theAddress.StateISOCode__c
    ,'country' => theAddress.Country__c
    ,'countryCode' => theAddress.CountryISOCode__c
};
```

global virtual Map<String,Object> fetchSubscriptions(Map<String,Object> inputData)

The fetchSubscriptions method is called from MyAccount -> My Subscriptions to return the list of subscription for the current user.

The OOTB implementation uses the current context to determine which subscriptions to return. The only filtering present is based in the page configuration MA.filteff. The returned data is constructed using the convertSubscriptionToMap utility method. In addition to the data returned by the utility call, the Map<String,Object> for the subscription should contain the key 'canCancel' which determines if the given subscription can be cancelled or not. For each subscription the original product ordered is added as the field 'subProduct' which is then produced by:

```
return new Map<String,Object>{
     'sfid' => theProd.id
    ,'name' => theProd.name
    ,'sku' => theProd.sku
    ,'productType' => theProd.ProductType
    ,'productStatus' => theProd.ProductStatus
    ,'shortDesc' => theProd.shortDesc
    ,'longDesc' => theProd.longDesc
};
```

Where the name, short description, and long description follow the normal CloudCraze localization rules.

Inputs

None

Outputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTIONS

The list of subscription data to display to the user as a List<Map<String,Object>>.

global virtual Map<String,Object> cancelSubscription(Map<String,Object> inputData)

This method is called from MyAccount when a subscription is cancelled.

The OOTB implementation changes the ccrz__SubscriptionStatus__c field of the subscription to 'Cancelled' and updates the subscription.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTIONID

The String id of the subscription to be cancelled.

Outputs

None

global virtual Map<String,Object> fetchSubscription(Map<String,Object> inputData)

This method is called from the subscription detail page and is used to pull the full subscription object.

The OOTB implementation returns the data as a Map<String,Object>. In addition to the data returned by the convertSubscriptionToMap method the map will contain 'canCancel', a List<Map<String,Object>> of all child subscriptions on the key 'subChildren', the 'subProduct' data (see fetchSubscriptions), any placed order data (see below) as 'orderHistory', as well as any transaction payment associated with the subscription as 'installmentPayments', see ccrz.cc_hk_Payment.convertTxnPaymentToMap.

```
return new Map<String,Object>{
     'sfid' => theOrder.Id
    ,'name' => theOrder.Name
    ,'orderNumber' => theOrder.OrderNumber__c
    ,'orderStatus' => theOrder.OrderStatus__c
    ,'orderDate' => (null==theOrder.OrderDate__c)?'':theOrder.
OrderDate__c.format()
    ,'orderSubtotal' => theOrder.SubtotalAmount__c
    ,'orderTotal' => theOrder.TotalAmount__c
};
```

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTIONID

The Salesforce id of the subscription to display.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The subscription as a Map<String,Object>.

global virtual Map<String,Object> updateSubscriptionPayment(Map<String,Object> inputData)

Updates the stored payment associated with a subscription.

The OOTB implementation sets the ccrz__StoredPayment__c field to the passed stored payment.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTIONID

The Salesforce id of the subscription.

ccrz.cc_hk_Subscriptions.PARAM_STOREDPAYMENTID

A JSON serialized representation of the stored payment. The JSON object should contain a field with the value of ccrz.cc_hk_Subscriptions. PARAM_STOREDPAYMENTID.

```
{
        'storedPaymentId' : '12345678'
}
```

Outputs

ccrz.cc_hk_Subscriptions.PARAM_

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The updated subscription as a Map<String,Object>

global virtual Map<String,Object> onOrder(Map<String,Object> inputData)

The onOrder method is called during checkout and is used to convert selected product with subscriptions to subscription records. Specifically, for all products on the cart that have selected ccrz__E_ProdSubTerm__c records this method creates corresponding ccrz__E_Subscription__c records. This call occurs after the order has been created and is part of the overall order transaction. In other words, any exceptions thrown during this process will cause all DML changes to rollback.

The OOTB implementation reads all created ccrz__E_OrderItem__c records and for each with a corresponding ccrz__E_ProdSubTerm__c record creates a new ccrz__E_Subscription__c record. Because the ccrz__E_OrderItem__c only has the price of the first billing cycle, valid modifier ccrz__E_ProdSubTerm__c are queried via the ***fetchSubProdTerms*** method of this hook which converts them to a Map<String, Object>. They are then priced using the ccrz.cc_hk_Pricing class and passed through ***filterSubscriptionOptions.*** Finally the ccrz__E_ProdSubTerm__c's in the list are passed in to ***createSubscription*** where a new ccrz__E_Subscription__c is created for each valid one.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_ORDER_ID

The Salesforce id of the newly created ccrz__E_Order__c.

ccrz.cc_hk_Subscriptions.PARAM_TRANSACTION_DATA

The parsed payment data passed as part of the checkout. The OOTB implementation assumes this is a ccrz.cc_hk_Payment. TransactionPaymentParams. The transaction payment data should contain the stored payment id to be associated with the recurring installment payments.

Outputs

None

global virtual Map<String,Object> calculateTargetDate(final Map<String,Object> inputData)

This method is used to calculate next target dates for subscription calculation. The next date is based on the frequency UOM and frequency (order or installment). If an invalid frequency UOM is passed then a ccrz.cc_hk_Subscriptions.DateUOMNotSpecifiedException is thrown.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_FREQUENCY_UOM

The frequency UOM used to calculate the next date.

ccrz.cc_hk_Subscriptions.PARAM_FREQUENCY

The frequency used to calculate the next date.

ccrz.cc_hk_Subscriptions.PARAM_COUNT

The count used to calculate the next date. Passing 1 would calculate the next date.

ccrz.cc_hk_Subscriptions.PARAM_START_DATE

The base date to use when calculating the date.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_TARGET_DATE

The next date.

global virtual Map<String,Object> createSubscription(Map<String,Object> inputData)

This method is called by onOrder for each subscription and child modifier to create the actual ccrz__E_Subscription__c object based on the order. This method is called within a loop so no DML or additional SOQL should take place within this method.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_ORDER

The newly created ccrz__E_Order__c object. OOTB the fields pulled are: ccrz__Storefront__c, ccrz__OrderDate__c, ccrz__CurrencyISOCode__c, ccrz__BillTo__c, ccrz__Account__c, ccrz__EffectiveAccountId__c.

ccrz.cc_hk_Subscriptions.PARAM_ORDER_ITEM

The ccrz__E_OrderItem__c object associated with this subscription. The fields pulled are: ccrz__Price__c, ccrz__SubAmount__c, ccrz__RecurringPrice__c, ccrz__RecurringPriceSubAmt__c, ccrz__Quantity__c.

ccrz.cc_hk_Subscriptions.PARAM_SUB_PROD_TERM

The ccrz__E_SubProdTerm__c object associated with the subscription being created.

ccrz.cc_hk_Subscriptions.PARAM_SPT_PARENT

If a modifier subscription is being created this is the parent ccrz__E_SubProdTerm__c record.

ccrz.cc_hk_Subscriptions.PARAM_SPT_ID

The Salesforce id of the stored payment tied to the subscription.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION_OBJ

A newly instantiated, but not inserted, ccrz__E_Subscription__c.

global virtual Map<String,Object> getSubscriptionProcessQuery(Map<String,Object> inputData)

This method is used by the OOTB subscription processing batch job to determine the overall scope of the subscriptions to be processed. The method returns a Salesforce Database.QueryLocator.

The OOTB query fields returned are:

```
 Id
,Name
,OwnerId
,ccrz__Account__c
,ccrz__AutoRenew__c
,ccrz__CCOrder__c
,ccrz__CCOrderItem__c
,ccrz__CurrencyISOCode__c
,ccrz__EffectiveAccountID__c
,ccrz__IgnoreInstallmentFrequencyFields__c
,ccrz__InstallmentAddress__c
,ccrz__InstallmentCountRemaining__c
,ccrz__InstallmentLastDate__c
,ccrz__InstallmentNextDate__c
,ccrz__InstallmentStatus__c
,ccrz__OrderAddress__c
,ccrz__OrderCountRemaining__c
,ccrz__OrderLastDate__c
,ccrz__OrderNextDate__c
,ccrz__OrderStatus__c
,ccrz__ParentSubscription__c
,ccrz__Quantity__c
,ccrz__RecurringPrice__c
,ccrz__RecurringPriceSubAmt__c
,ccrz__Sequence__c
,ccrz__SPT__c
,ccrz__SPTInstallmentCount__c
,ccrz__SPTInstallmentFrequency__c
,ccrz__SPTInstallmentFrequencyUOM__c
,ccrz__SPTOrderCount__c
,ccrz__SPTOrderFrequency__c
,ccrz__SPTOrderFrequencyUOM__c
,ccrz__StoredPayment__c
,ccrz__Storefront__c
,ccrz__SubscriptionId__c
,ccrz__SubscriptionProduct__c
,ccrz__SubscriptionStartDate__c
```

```
,ccrz__SubscriptionStatus__c
,ccrz__User__c
,ccrz__CCOrder__r.ccrz__Account__c
,ccrz__CCOrder__r.ccrz__Contact__c
,ccrz__CCOrder__r.ccrz__User__c
,ccrz__CCOrder__r.ccrz__EffectiveAccountID__c
,ccrz__InstallmentAddress__r.Name
,ccrz__InstallmentAddress__r.ccrz__AddressFirstline__c
,ccrz__InstallmentAddress__r.ccrz__AddressSecondline__c
,ccrz__InstallmentAddress__r.ccrz__AddressThirdline__c
,ccrz__InstallmentAddress__r.ccrz__AddrReadOnly__c
,ccrz__InstallmentAddress__r.ccrz__City__c
,ccrz__InstallmentAddress__r.ccrz__CompanyName__c
,ccrz__InstallmentAddress__r.ccrz__DaytimePhone__c
,ccrz__InstallmentAddress__r.ccrz__Email__c
,ccrz__InstallmentAddress__r.ccrz__FirstName__c
,ccrz__InstallmentAddress__r.ccrz__HomePhone__c
,ccrz__InstallmentAddress__r.ccrz__LastName__c
,ccrz__InstallmentAddress__r.ccrz__MailStop__c
,ccrz__InstallmentAddress__r.ccrz__MiddleName__c
,ccrz__InstallmentAddress__r.ccrz__Partner_Id__c
,ccrz__InstallmentAddress__r.ccrz__PostalCode__c
,ccrz__InstallmentAddress__r.ccrz__ShippingComments__c
,ccrz__InstallmentAddress__r.ccrz__State__c
,ccrz__InstallmentAddress__r.ccrz__StateISOCode__c
,ccrz__InstallmentAddress__r.ccrz__Country__c
,ccrz__InstallmentAddress__r.ccrz__CountryISOCode__c
,ccrz__OrderAddress__r.Name
,ccrz__OrderAddress__r.ccrz__AddressFirstline__c
,ccrz__OrderAddress__r.ccrz__AddressSecondline__c
,ccrz__OrderAddress__r.ccrz__AddressThirdline__c
,ccrz__OrderAddress__r.ccrz__AddrReadOnly__c
,ccrz__OrderAddress__r.ccrz__City__c
,ccrz__OrderAddress__r.ccrz__CompanyName__c
,ccrz__OrderAddress__r.ccrz__DaytimePhone__c
,ccrz__OrderAddress__r.ccrz__Email__c
,ccrz__OrderAddress__r.ccrz__FirstName__c
,ccrz__OrderAddress__r.ccrz__HomePhone__c
,ccrz__OrderAddress__r.ccrz__LastName__c
,ccrz__OrderAddress__r.ccrz__MailStop__c
,ccrz__OrderAddress__r.ccrz__MiddleName__c
,ccrz__OrderAddress__r.ccrz__Partner_Id__c
,ccrz__OrderAddress__r.ccrz__PostalCode__c
,ccrz__OrderAddress__r.ccrz__ShippingComments__c
,ccrz__OrderAddress__r.ccrz__State__c
,ccrz__OrderAddress__r.ccrz__StateISOCode__c
,ccrz__OrderAddress__r.ccrz__Country__c
,ccrz__OrderAddress__r.ccrz__CountryISOCode__c
,ccrz__StoredPayment__r.ccrz__Account__c
,ccrz__StoredPayment__r.ccrz__SubaccountNumber__c
```

```
,ccrz__StoredPayment__r.ccrz__AccountNumber__c
,ccrz__StoredPayment__r.ccrz__AccountType__c
,ccrz__StoredPayment__r.ccrz__Default__c
,ccrz__StoredPayment__r.ccrz__DisplayName__c
,ccrz__StoredPayment__r.ccrz__EffectiveAccountID__c
,ccrz__StoredPayment__r.ccrz__Enabled__c
,ccrz__StoredPayment__r.ccrz__EndDate__c
,ccrz__StoredPayment__r.ccrz__ExpMonth__c
,ccrz__StoredPayment__r.ccrz__ExpYear__c
,ccrz__StoredPayment__r.ccrz__Name__c
,ccrz__StoredPayment__r.ccrz__PaymentType__c
,ccrz__StoredPayment__r.ccrz__ReadOnly__c
,ccrz__StoredPayment__r.ccrz__Sequence__c
,ccrz__StoredPayment__r.ccrz__StartDate__c
,ccrz__StoredPayment__r.ccrz__Storefront__c
,ccrz__StoredPayment__r.ccrz__Token__c
,(
        SELECT
         Id
        ,Name
        ,OwnerId
        ,ccrz__Account__c
        ,ccrz__AutoRenew__c
        ,ccrz__CCOrder__c
        ,ccrz__CCOrderItem__c
        ,ccrz__CurrencyISOCode__c
        ,ccrz__EffectiveAccountID__c
        ,ccrz__IgnoreInstallmentFrequencyFields__c
        ,ccrz__InstallmentAddress__c
        ,ccrz__InstallmentCountRemaining__c
        ,ccrz__InstallmentLastDate__c
        ,ccrz__InstallmentNextDate__c
        ,ccrz__InstallmentStatus__c
        ,ccrz__OrderAddress__c
        ,ccrz__OrderCountRemaining__c
        ,ccrz__OrderLastDate__c
        ,ccrz__OrderNextDate__c
        ,ccrz__OrderStatus__c
        ,ccrz__ParentSubscription__c
        ,ccrz__Quantity__c
        ,ccrz__RecurringPrice__c
        ,ccrz__RecurringPriceSubAmt__c
        ,ccrz__Sequence__c
        ,ccrz__SPT__c
        ,ccrz__SPTInstallmentCount__c
        ,ccrz__SPTInstallmentFrequency__c
        ,ccrz__SPTInstallmentFrequencyUOM__c
        ,ccrz__SPTOrderCount__c
        ,ccrz__SPTOrderFrequency__c
        ,ccrz__SPTOrderFrequencyUOM__c
```

```
        ,ccrz__StoredPayment__c
        ,ccrz__Storefront__c
        ,ccrz__SubscriptionId__c
        ,ccrz__SubscriptionProduct__c
        ,ccrz__SubscriptionStartDate__c
        ,ccrz__SubscriptionStatus__c
        ,ccrz__User__c
        FROM ccrz__CCSubscriptions__r
        WHERE'
                (ccrz__InstallmentCountRemaining__c > 0 AND
ccrz__InstallmentCountRemaining__c != NULL)'
                ORDER BY ccrz__Sequence__c ASC LIMIT 1'
)
```

Depending on the particular context the subscriptions are then filtered as:

For Order only processing:

```
WHERE
    ccrz__SubscriptionStatus__c = \'Active\' AND ccrz__OrderNextDate__c
<= :dtNow
AND ccrz__Storefront__c = :theStorefront
AND ccrz__OrderCountRemaining__c > 0
AND
   (
        ccrz__IgnoreInstallmentFrequencyFields__c != TRUE
     OR ccrz__InstallmentCountRemaining__c < 1
     OR ccrz__InstallmentCountRemaining__c = NULL
   )
AND ccrz__ParentSubscription__c = NULL
```

For Installment only processing:

```
WHERE
    ccrz__SubscriptionStatus__c = \'Active\' AND
ccrz__InstallmentNextDate__c <= :dtNow
AND ccrz__Storefront__c = :theStorefront
AND ccrz__InstallmentCountRemaining__c > 0
AND
   (
         ccrz__IgnoreInstallmentFrequencyFields__c != TRUE
      OR ccrz__OrderCountRemaining__c < 1
      OR ccrz__OrderCountRemaining__c = NULL
   )
AND ccrz__ParentSubscription__c = NULL
```

For Order and Installment combined processing:

```
WHERE
    ccrz__SubscriptionStatus__c = \'Active\' AND ccrz__OrderNextDate__c
<= :dtNow
AND ccrz__Storefront__c = :theStorefront
AND ccrz__OrderCountRemaining__c > 0
AND ccrz__InstallmentCountRemaining__c > 0
AND ccrz__IgnoreInstallmentFrequencyFields__c = TRUE
AND ccrz__ParentSubscription__c = NULL
```

Note that 'theStorefront' and 'dtNow' are local variables to getSubscriptionProcessQuery.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR

The context of the subscriptions being processed. Will be one of:

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR_COMBINED_ONLY - Process those subscriptions where ignore installment frequency is true and both an order and installment are being processed as part of the same transaction.

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR_INSTALL_ONLY - Process those subscriptions where ignore installment frequency is false and/or there is an installment to process.

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR_ORDER_ONLY - Process those subscriptions where ignore installment frequency is false and/or there is an order to process.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_PROCESS_SUBSCRIPTIONS_LOCATOR

The Database.QueryLocator that defines the scope of the subscriptions to process.

global virtual Map<String,Object> preProcessSubscriptions(Map<String,Object> inputData)

This method is called by the OOTB processSubscriptions method and is used to collect original values for the order and installment counts and the order and installment next dates. This data is used to restore the original values to subscriptions that fail to process successfully.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_PROCESS_SUBSCRIPTIONS

The List<ccrz__E_Subscription__c> of subscriptions being processed.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_ORIG_SUBDATA

A Map<String,Object> of the data to be restored keyed by the subscription Salesforce id. Each value contains the original values for each subscription as a Map<String,Object> where the keys are the field names, e.g. ccrz__OrderCountRemaining__c.

global virtual Map<String,Object> processSubscriptions(Map<String,Object> inputData)

This is the main subscription process method called by the OOTB batch job.

The OOTB implementation first calls preProcessSubscriptions, then depending on the context being processed calls the appropriate method. Once all processing is complete the method calls postProcessSubscriptions.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR

The context of the subscriptions being processed. Will be one of:

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR_COMBINED_ONLY - The method processRecurrCombined will be called.

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR_INSTALL_ONLY - The method processRecurrInstallments will be called.

ccrz.cc_hk_Subscriptions.PARAM_CONTEXT_RECURR_ORDER_ONLY - The method processRecurrInstallments will be called.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

None

global virtual Map<String,Object> postProcessSubscriptions(Map<String,Object> inputData)

This method is called by the OOTB processSubscriptions method.

The OOTB implementation first calls any configured ccrz.ccPaymentProcessor implementations for any successful installment subscriptions, calling the postProcess method on the ccrz.ccPaymentProcessor. Next, the method calls postProcess for any failed transaction or subscriptions that have a configured ccrz.ccPaymentProcessor.

Finally the method calls processRecurrErrors.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_PAY_PROCESSORS

A Map<String,Object> of instantiated ccrz.ccPaymentProcessor objects keyed by the concatenation of storefront and the account type (payment method/type). For example, 'DefaultStorePO'. Only used when installment payments are present.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_TXN

The List<ccrz__E_TransactionPayment__c> of created transaction payments for installment. Note that this list will be the same length and may be indexed the same as the good subscriptions list. Only used when installment payments are present.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS

The List<ccrz__E_Subscription__c> of processed subscriptions. These are the top level subscriptions only. The child modifiers are not passed in this list. Only used when installment payments are present.

ccrz.cc_hk_Subscriptions.PARAM_BAD_SUBS

The Map<String,Object> containing subscription data that failed to process. Each entry is keyed by the subscription id and each value contains a Map<String,Object>. The key ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION contains the ccrz__E_Subscription__c object that failed to process. In addition the subscription failed to process due to an exception then the key ccrz.cc_hk_Subscriptions.PARAM_EXCEPTION will contain the exception. If the subscription failed to update or a record insert failed related to the subscription then the key ccrz.cc_hk_Subscriptions.PARAM_SAVERESULT will contain the Database.SaveResult that caused the failure.

ccrz.cc_hk_Subscriptions.PARAM_SUB_PAYDATA

The original payment data as processed for installment subscriptions. See processRecurrInstallments for how this data gets constructed.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

None

global virtual Map<String,Object> processRecurrOrders(Map<String,Object> inputData)

This method processes subscriptions with recurring orders. No payment handling should be done in this method.

The OOTB implementation first creates new billing and shipping address records (ccrz__E_ContactAddr__c) for each new order being created if those records are present on the original order. The method recurrOrderAddrInfo is called to create the address information.

Once the new instances of ccrz__E_ContactAddr__c have been created the implementation calls Database.insert on the instances and notes any failures.

Next, for each subscription the method recurrOrder is called to create the new Order object.

Once the new instances of ccrz__E_Order__c and ccrz__E_OrderItem__c have been created the implementation calls Database.insert on the instances and notes any failures.

Finally, the subscription order count is decremented and the order next date is calculated and updated. The method calculateTargetDate is used to calculate the next target date.

The implementation calls Database.update on all subscriptions to be updated and notes any failures.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_PROCESS_SUBSCRIPTIONS

The List<ccrz__E_Subscription__c> of subscriptions being processed.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS

The List<ccrz__E_Subscription__c> of subscriptions that were successfully processed. This list matches in length and index the PARAM_GOOD_ORDERS list.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_ORDERS

The List<ccrz__E_Order__c> of newly created orders.

ccrz.cc_hk_Subscriptions.PARAM_BAD_SUBS

The Map<String,Object> containing subscription data that failed to process. Each entry is keyed by the subscription id and each value contains a Map<String,Object>. The key ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION contains the ccrz__E_Subscription__c object that failed to process. In addition the subscription failed to process due to an exception then the key ccrz.cc_hk_Subscriptions.PARAM_EXCEPTION will contain the exception. If the subscription failed to update or a record insert failed related to the subscription then the key ccrz.cc_hk_Subscriptions.PARAM_SAVERESULT will contain the Database.SaveResult that caused the failure.

global virtual Map<String,Object> parsePaymentData(Map<String,Object> inputData)

This method is called from the processRecurrInstallments method for each subscription with an associated stored payment. The method is used to prepare the payment data and put it in a form that is common to the ccrz.ccPaymentProcessor implementation.

The OOTB implementation creates a new ccrz.cc_hk_Payment.TransactionPaymentParams object from the stored payment data associated with the subscription.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The ccrz__E_Subscription__c object being processed.

Outputs

ccrz.cc_hk_Payment.PARAM_TRANSACTION_PROCESSED_DATA

The returned parsed data.

global virtual Map<String,Object> createTransactionPayment(Map<String,Object> inputData)

This method should create a new ccrz__E_TransactionPayment__c object (but not save it) for a subscription where installment billing is used.

The OOTB implementation creates a new ccrz__E_TransactionPayment__c object from the subscription and parsed payment data. Specifically:

```
final E_Subscription__c theSub = (E_Subscription__c)inputData.get
(PARAM_SUBSCRIPTION);
final cc_hk_Payment.TransactionPaymentParams txnParams =
    (cc_hk_Payment.TransactionPaymentParams)inputData.get(cc_hk_Payment.
PARAM_TRANSACTION_PROCESSED_DATA);

E_Subscription__c amtSub = theSub;

final List<E_Subscription__c> mainChildSub = theSub.CCSubscriptions__r;
if((null!=mainChildSub) && !mainChildSub.isEmpty()){
    amtSub = mainChildSub[0];
}


final E_TransactionPayment__c txnPay = new E_TransactionPayment__c(
             CCSubscription__c = theSub.Id
            ,CurrencyISOCode__c = theSub.CurrencyISOCode__c
            ,Storefront__c = theSub.Storefront__c
            ,Account__c = theSub.Account__c
            ,User__c = theSub.User__c
            ,OwnerId = theSub.OwnerId
            ,Amount__c = amtSub.RecurringPriceSubAmt__c
            ,RequestAmount__c = amtSub.RecurringPriceSubAmt__c
            ,AccountNumber__c = txnParams.accountNumber
            ,AccountType__c = txnParams.accountType
            ,Comments__c = txnParams.comments
            ,ExpirationMonth__c = txnParams.expirationMonth
            ,ExpirationYear__c = txnParams.expirationYear
            ,PaymentType__c = txnParams.paymentType
            ,StoredPayment__c = txnParams.storedPaymentId
            ,SubAccountNumber__c = txnParams.subAccountNumber
            ,Token__c = txnParams.token
            ,TransactionCode__c = txnParams.transactionCode
            ,TransactionSubcode__c = txnParams.transactionSubcode
            ,TransactionType__c = txnParams.transactionType
            ,VerificationCode__c = txnParams.verificationCode
        );


return new Map<String,Object>{
    PARAM_TXN_PAYMENT => txnPay
};
```

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The subscription being processed.

ccrz.cc_hk_Payment.PARAM_TRANSACTION_PROCESSED_DATA

The parsed payment data from the parsePaymentData call.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_TXN_PAYMENT

The new ccrz__E_TransactionPayment__c object.

global virtual Map<String,Object> processRecurrInstallments(Map<String,Object> inputData)

This method is call to process recurring installment payments. This method is ultimately responsible for updating the subscription installment count and installment next date.

The default implementation first separates the subscriptions into two groups. Those with stored payments and those without. For those subscriptions with stored payments the parsePaymentData method is first called. Next the implementation determines if there is a ccrz.ccPaymentProcessor for the given account type and then calls preProcess on that processor. The method ccrz.cc_hk_Payment.getPaymentProcessor is used to determine the payment processor. During this initial process callouts are allowed.

Next the implementation processes the installment subscriptions without a stored payment. For these subscriptions the installment date and count are updated. Note that no transaction payments are created. Once the subscription data has been updated the subscriptions are updated using Database.update and any errors are stored off as bad subscriptions.

Next the implementation processes the installment subscriptions that include stored payments. First, the new transaction payment instances are created using the createTransactionPayment method. The new transactions are save off but not yet inserted.

The implementation then inserts the new ccrz__E_TransactionPayment__c instances using Database.insert. Any failed inserts are noted.

For any subscriptions where a ccrz__E_TransactionPayment__c was successfully inserted the implementation next calls ccrz.ccPaymentProcessor.process for each subscription. The process method is called passing the new transaction payment object, the subscription, the parsed payment data, the stored payment, and a context indicating the process is being called from the subscription installment process. Any errors are flagged.

During the handling of the process calls the subscription is also updated with the next installment date and the installment count is decremented. Any errors are flagged.

The subscriptions are next updated via Database.update and any errors are noted.

Finally, any ccrz__E_TransactionsPayment__c objects that were created but where the subscription for that payment subsequently had an error are deleted. Specifically, if a payment was created but the subscription processing failed, we do not leave the new transaction payment in the system.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_PROCESS_SUBSCRIPTIONS

The List<ccrz__E_Subscription__c> of subscriptions being processed.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS_NO_TXN

The List<ccrz__E_Subscription__c> that were updated but for which there was no stored payment.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS

The List<ccrz__E_Subscription__c> that were updated that had a stored payment and for which a new ccrz__E_TransactionPayment__c was created. The size and index of this list matches PARAM_GOOD_TXN.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_TXN

The List<ccrz__E_TransactionPayment__c> that were created for the subscriptions with stored payments.

ccrz.cc_hk_Subscriptions.PARAM_BAD_SUBS

The Map<String,Object> containing subscription data that failed to process. Each entry is keyed by the subscription id and each value contains a Map<String,Object>. The key ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION contains the ccrz__E_Subscription__c object that failed to process. In addition the subscription failed to process due to an exception then the key ccrz.cc_hk_Subscriptions.PARAM_EXCEPTION will contain the exception. If the subscription failed to update or a record insert failed related to the subscription then the key ccrz.cc_hk_Subscriptions.PARAM_SAVERESULT will contain the Database.SaveResult that caused the failure.

ccrz.cc_hk_Subscriptions.PARAM_PAY_PROCESSORS

Any looked up ccrz.ccPaymentProcessors as a Map<String,Object> keyed by Storefront concatenated with the account type. E.g. DefaultStorePO.

ccrz.cc_hk_Subscriptions.PARAM_SUB_PAYDATA

The results of the preProcess calls to ccrz.ccPaymentProcessor for each subscription keyed by the Salesforce id of the ccrz__E_Subscription__c record.

global virtual Map<String,Object> processRecurrCombined(Map<String,Object> inputData)

This method is called for subscriptions where the order and installment are part of the same transaction. This occurs when the ignore installment frequency flag is true.

The default implementation first calls processRecurrInstallments.

For subscriptions that were successful without transactions the implementation then calls processRecurrOrders.

For subscriptions that were successful with transactions the implementation calls processRecurrOrders.

Note that for each step the set of failed subscriptions is noted.

Finally, for any subscriptions where a transaction was created but for which the order recurrence failed the transactions are deleted.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_PROCESS_SUBSCRIPTIONS

The List<ccrz__E_Subscription__c> of subscriptions being processed.

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront of the subscriptions to process.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_GOOD_ORDERS

The List<ccrz__E_Order__c> of newly created orders.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS_NO_TXN

The List<ccrz__E_Subscription__c> that were updated but for which there was no stored payment.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS

The List<ccrz__E_Subscription__c> that were updated that had a stored payment and for which a new ccrz__E_TransactionPayment__c was created. The size and index of this list matches PARAM_GOOD_TXN.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_TXN

The List<ccrz__E_TransactionPayment__c> that were created for the subscriptions with stored payments.

ccrz.cc_hk_Subscriptions.PARAM_PAY_PROCESSORS

Any looked up ccrz.ccPaymentProcessors as a Map<String,Object> keyed by Storefront concatenated with the account type. E.g. DefaultStorePO.

ccrz.cc_hk_Subscriptions.PARAM_SUB_PAYDATA

The results of the preProcess calls to ccrz.ccPaymentProcessor for each subscription keyed by the Salesforce id of the ccrz__E_Subscription__c record.

ccrz.cc_hk_Subscriptions.PARAM_BAD_SUBS

The Map<String,Object> containing subscription data that failed to process. Each entry is keyed by the subscription id and each value contains a Map<String,Object>. The key ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION contains the ccrz__E_Subscription__c object that failed to process. In addition the subscription failed to process due to an exception then the key ccrz.cc_hk_Subscriptions.PARAM_EXCEPTION will contain the exception. If the subscription failed to update or a record insert failed related to the subscription then the key ccrz.cc_hk_Subscriptions.PARAM_SAVERESULT will contain the Database.SaveResult that caused the failure.

global virtual Map<String,Object> processRecurrErrors(Map<String,Object> inputData)

This method processes any subscriptions that failed to process.

The default implementation restores the original values for the subscription. See preProcessSubscriptions to see the data captured.

The default implementation then sets the ccrz__SubscriptionStatus__c field to 'Error' and puts the failure message into the field ccrz__LastRecurringFailure__c.

The subscriptions are then updated using Database.update.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_BAD_SUBS

The Map<String,Object> containing subscription data that failed to process. Each entry is keyed by the subscription id and each value contains a Map<String,Object>. The key ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION contains the ccrz__E_Subscription__c object that failed to process. In addition the subscription failed to process due to an exception then the key ccrz.cc_hk_Subscriptions.PARAM_EXCEPTION will contain the exception. If the subscription failed to update or a record insert failed related to the subscription then the key ccrz.cc_hk_Subscriptions.PARAM_SAVERESULT will contain the Database.SaveResult that caused the failure.

ccrz.cc_hk_Subscriptions.PARAM_ORIG_SUBDATA

The original subscription data as a Map<String,Object> keyed by the subscription Salesforce id.

Outputs

None

global virtual Map<String,Object> recurrOrderAddrInfo(Map<String,Object> inputData)

This method is called to create the new ccrz__E_ContactAddr__c objects to be inserted and associated to the new ccrz__E_Order__c.

The default implementation copies the data from the original address records (if present) and sets the OwnerId of the newly created objects to the OwnerId of the subscription.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The subscription being processed.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_SHIP_ADDR

The new ccrz__E_ContactAddr__c instance for the new order shipping address.

ccrz.cc_hk_Subscriptions.PARAM_BILL_ADDR

The new ccrz__E_ContactAddr__c instance for the new order billing address.

global virtual Map<String,Object> recurrOrder(Map<String,Object> inputData)

This method is called during the recurring order process to create a new ccrz__E_Order__c instance and new ccrz__E_OrderItem__c instance.

> CC orders generated by this method display in the Subscription Detail page, but are filtered out of the My Orders (Order History) page of My Account.

The default implementation copies the order data from the original order and subscription objects. Specifically:

```
ccrz__E_Subscription__c theSub = (ccrz__E_Subscription__c)inputData.get
(PARAM_SUBSCRIPTION);
List<ccrz__E_OrderItem__c> minItems = (List<ccrz__E_OrderItem__c>)
inputData.get(PARAM_MIN_ITEMS);

final String ordId = (String)theSub.Id+String.valueOf(System.
currentTimeMillis());
final String ordItId = ordId+String.valueOf(System.currentTimeMillis());
ccrz__E_Order__c theOrder = new ccrz__E_Order__c(
         ccrz__CCSubscription__c = theSub.Id
        ,ccrz__CurrencyISOCode__c = theSub.ccrz__CurrencyISOCode__c
        ,ccrz__Order__c = theSub.ccrz__CCOrder__c
        ,ccrz__OrderDate__c = Date.today()
        ,ccrz__OrderStatus__c = 'Order Submitted'
        ,ccrz__Storefront__c = theSub.ccrz__Storefront__c
        ,ccrz__TaxAmount__c = 0.0
        ,ccrz__TotalDiscount__c = 0.0
        ,ccrz__TotalSurcharge__c = 0.0
        ,ccrz__Account__c = theSub.ccrz__CCOrder__r.ccrz__Account__c
        ,ccrz__Contact__c = theSub.ccrz__CCOrder__r.ccrz__Contact__c
        ,ccrz__User__c = theSub.ccrz__CCOrder__r.ccrz__User__c
        ,ccrz__EffectiveAccountID__c = theSub.
ccrz__EffectiveAccountID__c
        ,OwnerId = theSub.OwnerId
        ,ccrz__OrderId__c = ordId
);

ccrz__E_ContactAddr__c theShipAddr = (ccrz__E_ContactAddr__c)inputData.
get(PARAM_SHIP_ADDR);
```

```
if((null!=theShipAddr)&&(null!=theShipAddr.Id)){
        theOrder.ccrz__ShipTo__c = theShipAddr.Id;
}

ccrz__E_ContactAddr__c theBillAddr = (ccrz__E_ContactAddr__c)inputData.
get(PARAM_BILL_ADDR);
if((null!=theBillAddr)&&(null!=theBillAddr.Id)){
        theOrder.ccrz__BillTo__c = theBillAddr.Id;
}

ccrz__E_OrderItem__c theOrderItem = new ccrz__E_OrderItem__c(
         ccrz__AbsoluteDiscount__c = 0.0
        ,ccrz__AdjustmentAmount__c = 0.0
        ,ccrz__SubProdTerm__c = theSub.ccrz__SPT__c
        ,ccrz__ItemStatus__c = 'Available'
        ,ccrz__OrderItemStatus__c = 'Order Submitted'
        ,ccrz__OrderLineType__c = 'Major'
        ,ccrz__OriginalQuantity__c = theSub.ccrz__Quantity__c
        ,ccrz__PercentDiscount__c = 0.0
        ,ccrz__Price__c = theSub.ccrz__RecurringPrice__c
        ,ccrz__Product__c = theSub.ccrz__SubscriptionProduct__c
        ,ccrz__Quantity__c = theSub.ccrz__Quantity__c
        ,ccrz__StoreId__c = theSub.ccrz__Storefront__c
        ,ccrz__SubAmount__c = theSub.ccrz__RecurringPriceSubAmt__c
        ,ccrz__Order__r = new ccrz__E_Order__c(ccrz__OrderId__c = ordId)
        ,ccrz__OrderItemId__c = ordItId
);

//Handle any minor items that need to be added to the order as part of
complex products
List<ccrz__E_OrderItem__c> minorItemsToAdd = new
List<ccrz__E_OrderItem__c>();
if(ccUtil.isNotEmpty(minItems)){
        for(ccrz__E_OrderItem__c minItem : minItems){
                minorItemsToAdd.add(new ccrz__E_OrderItem__c(
                         ccrz__AbsoluteDiscount__c = 0.0
                        ,ccrz__AdjustmentAmount__c = 0.0
                        ,ccrz__SubProdTerm__c = theSub.ccrz__SPT__c
                        ,ccrz__ItemStatus__c = 'Available'
                        ,ccrz__OrderItemStatus__c = 'Order Submitted'
                        ,ccrz__OrderLineType__c = 'Minor'
                        ,ccrz__OriginalQuantity__c = minItem.
ccrz__OriginalQuantity__c
                        ,ccrz__PercentDiscount__c = 0.0
                        ,ccrz__Price__c = 0.00
                        ,ccrz__Product__c = minItem.ccrz__Product__c
                        ,ccrz__Quantity__c = minItem.
ccrz__OriginalQuantity__c * theOrderItem.ccrz__Quantity__c
                        ,ccrz__StoreId__c = theSub.ccrz__Storefront__c
                        ,ccrz__SubAmount__c = 0.00
```

```
                        ,ccrz__Order__r = new ccrz__E_Order__c
(ccrz__OrderId__c = ordId)
                        ,ccrz__ParentOrderItem__r = new
ccrz__E_OrderItem__c(ccrz__OrderItemId__c = ordItId)
                ));
        }
}

return new Map<String,Object>{
         PARAM_ORDER=>theOrder
        ,PARAM_ORDER_ITEM=>theOrderItem
        ,PARAM_MINOR_ORDER_ITEMS=>minorItemsToAdd
};
```

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The subscription being processed.

ccrz.cc_hk_Subscriptions.PARAM_SHIP_ADDR

The new ccrz__E_ContactAddr__c instance for the new order shipping address.

ccrz.cc_hk_Subscriptions.PARAM_BILL_ADDR

The new ccrz__E_ContactAddr__c instance for the new order billing address.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_ORDER

The newly instantiated ccrz__E_Order__c.

ccrz.cc_hk_Subscriptions.PARAM_ORDER_ITEM

The newly instantiated ccrz__E_OrderItem__c.

ccrz.cc_hk_Subscriptions.PARAM_MINOR_ORDER_ITEMS

The newly instantiated ccrz__E_OrderItem__c that are the minor items for the newly created main order item representing the subscription item.

global virtual Map<String,Object> getSubscriptionRenewalQuery(Map<String,Object> inputData)

This method is called during the subscription renewal process to determine the scope of the subscriptions that are available for renewal. The method returns a Database.QueryLocator.

The OOTB implementation returns the same fields as getSubscriptionProcessQuery but uses different filtering. Specifically:

```
WHERE
    SubscriptionStatus__c = \'Active\'
AND Storefront__c = :theStorefront
AND
(
        OrderCountRemaining__c < 1
     OR OrderCountRemaining__c = NULL
)
AND
(
        InstallmentCountRemaining__c < 1
     OR InstallmentCountRemaining__c = NULL
)
AND ParentSubscription__c = NULL
```

Inputs

ccrz.cc_hk_Subscriptions.PARAM_STOREFRONT

The storefront for the subscriptions being processed.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_RENEWAL_SUBSCRIPTIONS_LOCATOR

The Database.QueryLocator representing the set of subscriptions to process.

global virtual Map<String,Object> renewAddrInfo(Map<String,Object> inputData)

This method is called by processRenewals if the subscription is set to be auto renewed.

The default implementation copies the data from the original address records (if present) and sets the OwnerId of the newly created objects to the OwnerId of the subscription.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION

The subscription being processed.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_SHIP_ADDR

The new ccrz__E_ContactAddr__c instance for the new subscription ship address.

ccrz.cc_hk_Subscriptions.PARAM_BILL_ADDR

The new ccrz__E_ContactAddr__c instance for the new subscription bill address.

global virtual Map<String,Object> renewSubscription(final Map<String,Object> inputData)

This method is called by the processRenewals method for each subscription to be renewed.

The OOTB implementation first clones the old subscription. The start date for the new subscription is set to be the latest date of either ccrz__OrderLastDate__c or ccrz__InstallmentLastDate__c.

The implementation next sets the counts for the new subscription based on the old subscription sub prod term data. Specifically:

```
newSub.ccrz__InstallmentCountRemaining__c = sourceSub.
ccrz__SPTInstallmentCount__c;
newSub.ccrz__InstallmentStatus__c          = ccrz.cc_hk_Subscriptions.
PARAM_STATUS_ACTIVE;
newSub.ccrz__OrderCountRemaining__c        = sourceSub.
ccrz__SPTOrderCount__c;
newSub.ccrz__OrderStatus__c                = ccrz.cc_hk_Subscriptions.
PARAM_STATUS_ACTIVE;
newSub.ccrz__SubscriptionStartDate__c      = sourceLast; //The latest of
ccrz__OrderLastDate__c or ccrz__InstallmentLastDate__c
newSub.ccrz__SubscriptionStatus__c         = ccrz.cc_hk_Subscriptions.
PARAM_STATUS_ACTIVE;
newSub.ccrz__SubscriptionId__c             = null;
newSub.ccrz__InstallmentLastDate__c        = null;
newSub.ccrz__InstallmentNextDate__c        = null;
newSub.ccrz__OrderLastDate__c              = null;
newSub.ccrz__OrderNextDate__c              = null;
```

Next the installment last and next, and the order last, and next dates are calculated. Note that if ignored installment frequency is set to true then only the order frequency fields will be used. The calculateTargetDate method is used to perform these calculations.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION_OBJ

The old subscription object.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION_OBJ

The new subscription object.

global virtual Map<String,Object> processRenewals(Map<String,Object> inputData)

This method is called by the subscription process batch job to handle any subscriptions that have expired. When the subscription is set to auto renew a new subscription will be created that carries forward the old subscription dates and the old subscription is expired. If the subscription is not auto renew the the subscription is simply expired.

The OOTB implementation first creates the new ccrz__E_ContactAddr__c instances for any subscriptions to be auto-renewed. During this process any subscriptions that are not auto-renewed are separated out into a different list.

The implementation next attempts to create the new ccrz__E_ContactAddr__c records using Database.insert. Any errors are noted.

The implementation next creates the top level subscriptions. Top level subscriptions are the main subscription objects (as opposed to the modifiers).

The implementation then inserts the new top level subscriptions using Database.insert. Any errors are noted.

Next the modifier child subscriptions are created as inserted following the same process as the top level subscriptions. Any errors are noted.

Next the subscriptions are updated to either 'Renewed' or 'Expired' depending on the value of the auto-renew flag.

Finally the subscriptions are updated using Database.update.

Any subscriptions that were created but for which some part of the process failed are deleted.

Inputs

ccrz.cc_hk_Subscriptions.PARAM_PROCESS_SUBSCRIPTIONS

The list of subscriptions to be processed.

Outputs

ccrz.cc_hk_Subscriptions.PARAM_BAD_SUBS

The Map<String,Object> containing subscription data that failed to process. Each entry is keyed by the subscription id and each value contains a Map<String,Object>. The key ccrz.cc_hk_Subscriptions.PARAM_SUBSCRIPTION contains the ccrz__E_Subscription__c object that failed to process. In addition the subscription failed to process due to an exception then the key ccrz.cc_hk_Subscriptions.PARAM_EXCEPTION will contain the exception. If the subscription failed to update or a record insert failed related to the subscription then the key ccrz.cc_hk_Subscriptions.PARAM_SAVERESULT will contain the Database.SaveResult that caused the failure.

ccrz.cc_hk_Subscriptions.PARAM_GOOD_SUBS

The List<ccrz__E_Subscription__c> of subscriptions that were updated.

Scheduling the Subscription Processor batch job

An example implementation of the schedulable apex job:

ccScheduledSubscriptionProcessor

```
global with sharing class ccScheduledSubscriptionProcessor implements
Schedulable{
    global void execute(SchedulableContext sc) {
        Database.executeBatch(new ccrz.cc_batch_SubscriptionProcessor
('DefaultStore',null,true));
    }
}
```

And then to run every 15 minutes (modify as necessary to run less frequently)

Anonymous Apex to schedule job

```
System.schedule('Job1', '0 0 * * * ?', new ccrz.
ccScheduledSubscriptionProcessor());
System.schedule('Job2', '0 15 * * * ?', new ccrz.
ccScheduledSubscriptionProcessor());
System.schedule('Job3', '0 30 * * * ?', new ccrz.
```

```
ccScheduledSubscriptionProcessor());
System.schedule('Job4', '0 45 * * * ?', new ccrz.
ccScheduledSubscriptionProcessor());
```