

SiteGenesis code changes

To integrate the new Afterpay cartridge, modifications must be made to your version of the SiteGenesis base cartridge files. A 3-way visual merge tool such as Meld (<https://meldmerge.org/>) can be useful for making these changes—if you’ve made considerable changes to the SiteGenesis base cartridge files for your own store.

Note: If you use SFRA, you can skip this page – there are no code changes needed.

1. File: **app_storefront_core/cartridge/scss/default/_afterpay.scss**

Copy the **_afterpay.scss** file from the **reference** folder of the v2.0 cartridge into **app_storefront_core/cartridge/scss/default/_afterpay.scss**

2. File: **app_storefront_controllers/cartridge/controllers/COBilling.js**

In the function `resetPaymentForms()`, add the highlighted line into the 3 locations:

```
var status = Transaction.wrap(function () {  
    var paymentMethod =  
    app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value;  
    if  
(app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(  
'PayPal')) {  
  
        app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();  
        app.getForm('billing').object.paymentMethods.bml.clearFormElement();  
  
        cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD_C  
REDIT_CARD));  
  
        cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD_B  
ML));  
    } else if  
(app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(  
PaymentInstrument.METHOD_CREDIT_CARD)) {  
        app.getForm('billing').object.paymentMethods.bml.clearFormElement();  
    }  
});
```

```

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD_B
ML));
        cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));
    } else if
(app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(
PaymentInstrument.METHOD_BML)) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

        if (!app.getForm('billing').object.paymentMethods.bml.ssn.valid) {
            return false;
        }

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD_C
REDIT_CARD));
        cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

    } else if
(app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(
'AFTERPAY') ||
app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('
CLEARPAY') ||
app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('
CASHAPPPAY')) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();
        app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD_C
REDIT_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD_B
ML));
        cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

    }
    // removes Afterpay PaymentInstrument

require('int_afterpay_sg/cartridge/scripts/checkout/afterpayHandlers').handleChange
dPaymentInstrument(paymentMethod);
    return true;
});

```

In the start() function, paste the highlighted code:

```

pageMeta.update({
    pageTitle: Resource.msg('billing.meta.pagetitle', 'checkout', 'SiteGenesis Checkout')
});
// Start of Afterpay

require('int_afterpay_sg/cartridge/scripts/checkout/afterpayHandlers').handleBillingStart();
// End of Afterpay
returnToForm(cart, params);

```

Also, in the same file, add highlighted lines to handlePaymentSection function:

```

function handlePaymentSelection(cart) {
    var afterpayBrandUtilities = AfterpayUtilities.brandUtilities;
    var selectedPaymentMethodID =
app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value;
    var result;

    if (empty(selectedPaymentMethodID)) {
        if (cart.getTotalGrossPrice() > 0) {
            result = {
                error: true
            };
        } else {
            result = {
                ok: true
            };
        }
    }

    if ((selectedPaymentMethodID == 'AFTERPAY' || selectedPaymentMethodID ==
'CLEARPAY') && !afterpayBrandUtilities.isAfterpayApplicable()) {
        result = {
            error: true
        };
    }

    // skip the payment handling if the whole payment was made using gift cert
    if (selectedPaymentMethodID.equals(PaymentInstrument.METHOD_GIFT_CERTIFICATE)) {
        result = {
            ok: true
        };
    }

    if

```

```

(empty(PaymentMgr.getPaymentMethod(selectedPaymentMethodID).paymentProcessor)) {
    result = {
        error: true,
        MissingPaymentProcessor: true
    };
}
if (!result) {
    result = app.getModel('PaymentProcessor').handle(cart.object,
selectedPaymentMethodID);
}
return result;
}

```

Also, in the same file, add highlighted lines to billing function:

```

function billing() {

    app.getForm('billing').handleAction({
        applyCoupon: function () {
            var couponCode = request.httpParameterMap.couponCode.stringValue ||
request.httpParameterMap.dwfrm_billing_couponCode.stringValue;

            // TODO what happened to this start node?
            app.getController('Cart').AddCoupon(couponCode);

            handleCoupon();
            return;
        },
        creditCardSelect: function () {
            updateCreditCardSelection();
            return;
        },
        paymentSelect: function () {
            // ToDo - pass parameter ?
            publicStart();
            return;
        },
        redeemGiftCert: function () {
            var status =
redeemGiftCertificate(app.getForm('billing').object.giftCertCode.htmlValue);
            if (!status.isError()) {
                returnToForm(app.getModel('Cart').get(), {
                    NewGCPaymentInstrument:
status.getDetail('NewGCPaymentInstrument')
                });
            } else {

```

```

        returnToForm(app.getModel('Cart').get());
    }

    return;
},
save: function () {
    Transaction.wrap(function () {
        var cart = app.getModel('Cart').get();

        if (!resetPaymentForms() || !validateBilling() ||
!handleBillingAddress(cart) // Performs validation steps, based upon the entered
billing address
        // and address options.
        ) {
            returnToForm(cart);
        } else {
            var handlePaymentSelectionResult =
handlePaymentSelection(cart); // Performs payment method specific checks, such as
credit card verification.

            if (handlePaymentSelectionResult.error){
                returnToForm(cart);
            }

            if (customer.authenticated &&
app.getForm('billing').object.billingAddress.addToAddressBook.value) {

app.getModel('Profile').get(customer.profile).addAddressToAddressBook(cart.getBilli
ngAddress());

            }

            // Mark step as fulfilled
            app.getForm('billing').object.fulfilled.value = true;

            } // A successful billing page will jump to the next checkout
step.

            if (!handlePaymentSelectionResult.redirect) {
                app.getController('COSummary').Start();
            }

            return;
        }
    });
},
selectAddress: function () {
    updateAddressDetails();
    return;
}
}

```

```
});  
}
```

3. File: **app_storefront_controllers/cartridge/controllers/COSummary.js**

In the start() function, remove the line:

```
app.getView(viewContext).render('checkout/summary/summary');
```

And paste in the following:

```
require("*/cartridge/scripts/checkout/afterpaySGCheckoutHelpers").disableSummaryFor  
Afterpay(cart, viewContext);
```

4. File: **app_storefront_controllers/cartridge/controllers/COPlaceOrder.js**

In the handlePayments() function, paste the following:

```
if (authorizationResult.not_supported || authorizationResult.error) {  
    return {  
        error: true  
    };  
}
```

With:

```
if (authorizationResult.not_supported || authorizationResult.error) {  
    return {  
        authorizeError : authorizationResult.authorizationResponse,  
        error: true  
    };  
}
```

Replace the code block:

```

if (handlePaymentsResult.error) {
  return Transaction.wrap(function () {
    OrderMgr.failOrder(order);
    return {
      error: true,
      PlaceOrderError: new Status(Status.ERROR, 'confirm.error.technical')
    };
  });
}

```

with:

```

if (handlePaymentsResult.error) {
  return Transaction.wrap(function () {
    OrderMgr.failOrder(order);
    return {
      error: true,
      afterpayOrderAuthorizeError: handlePaymentsResult.authorizeError,
      PlaceOrderError: new Status(Status.ERROR, 'confirm.error.technical')
    };
  });
}

```

5. File: **app_storefront_controllers/cartridge/controllers/Currency.js**

```

function setSessionCurrency() {
  var afterpayUtilities = require('*/cartridge/scripts/util/afterpayUtilities');
  var currencyMnemonic = request.httpParameterMap.currencyMnemonic.value;
  var locale = request.httpParameterMap.locale.value;
  var Response = require('*/cartridge/scripts/util/Response');
  var currency;

  if (currencyMnemonic) {
    currency = Currency.getCurrency(currencyMnemonic);
    if (currency) {
      session.setCurrency(currency);

      Transaction.wrap(function () {
        var currentCart = Cart.get();
        if (currentCart) {
          currentCart.updateCurrency();
          currentCart.calculate();
        }
      });
    }
  }
}

```

```

    }

    if (afterpayUtilities.sitePreferencesUtilities.isAfterpayEnabled() && locale) {
        afterpayUtilities.brandUtilities.initBrand(locale);
        var thresholdUtilities =
require('*/cartridge/scripts/util/thresholdUtilities');
        var brand = afterpayUtilities.brandUtilities.getBrand();
        thresholdUtilities.getThresholdAmounts(brand);
    }

    Response.renderJSON({
        success: true
    });
}

```

6. File: **app_storefront_controllers/cartridge/controllers/COShipping.js**

Near the bottom of function selectShippingMethod(), paste the highlighted code :

```

Transaction.wrap(function () {
    cart.updateShipmentShippingMethod(cart.getDefaultShipment().getID(),
request.httpParameterMap.shippingMethodID.stringValue, null,
applicableShippingMethods);
    cart.calculate();
});
// Start of Afterpay

require('int_afterpay_sg/cartridge/scripts/checkout/afterpayHandlers').handleShippingMethodUpdate();
// End of Afterpay
app.getView({
    Basket: cart.object
}).render('checkout/shipping/selectshippingmethodjson');

```


7. File:

app_storefront_core/cartridge/templates/default/components/header/htmlhead_UI.isml

Locate the following code:

```
<!--[if lt IE 9]>
<script src="{URLUtils.staticURL('/js/lib/html5.js')}"></script>
<![endif]-->
```

And add this code after it:

```
<isinclude url="{URLUtils.url('Afterpay-IncludeAfterpayLibrary')}" />
```

8. File:

app_storefront_core/cartridge/templates/default/checkout/billing/billing.isml

Locate the

```
<iscomment> ++++++billing address</iscomment>
```

and paste the code below after the comment as seen in the screenshot:

```
<iscomment>Start of Afterpay</iscomment>
<isset name="afterpayError"
  value="{!empty(pdikt.AfterpayApiError) ? pdikt.AfterpayApiError :
    (!empty(request.httpParameterMap.get('afterpay'))
      .stringValue ? request.httpParameterMap.get('afterpay').stringValue
      : null)}"
  scope="page" />
<isif condition="{!empty(afterpayError)}">
  <div class="error-form">
    <isprint value="{afterpayError}" encoding="off" />
  </div>
</isif>
<iscomment>End of Afterpay</iscomment>
```

9. File:

app_storefront_core/cartridge/templates/default/checkout/billing/payment_methods.isml

After the line:

```
<isinclude template="util/modules"/>
```

Paste the following code as show in the screenshot:

```
<isif condition="${pdict.OrderTotal > 0}">
  <iscomment>Start of Afterpay</iscomment>
  <isset name="afterpayOrdertotal" value="${pdict.OrderTotal}" scope="page"/>
  <isafterpaywidget pagetype="payment_methods" />
  <isif condition="${pdict.gcPITotal > 0}">
    <isset name="afterpayOrdertotal" value="${pdict.OrderTotal -
pdict.gcPITotal}" scope="page"/>
  </isif>
  <isscript>
    var sitePreferences =
require("*/cartridge/scripts/util/afterpayUtilities").sitePreferencesUtilities;
    var afterpayApplicable = sitePreferences.isAfterpayEnabled();
    var brandUtilities =
require("*/cartridge/scripts/util/afterpayUtilities").brandUtilities;
    brandUtilities.initBrand(pdict.CurrentRequest.locale);
    var apBrand = brandUtilities.getBrand();
    var AfterpayCOHelpers =
require('*/cartridge/scripts/checkout/afterpayCheckoutHelpers');
    var withinThreshold =
require('*/cartridge/scripts/util/thresholdUtilities').checkPriceThreshold(afterpay
Ordertotal);
    afterpayApplicable = afterpayApplicable ?
brandUtilities.isAfterpayApplicable() && withinThreshold.status &&
AfterpayCOHelpers.getCartData().apCartEligible : false;
    var cashAppEnabled = sitePreferences.isCashAppEnabled();
    var afterpayPaymentName = '';
    var mpid = withinThreshold.mpid;
    var countrycode = brandUtilities.getCountryCode();
  </isscript>
  <iscomment>End of Afterpay</iscomment>
```

After the line:

```
<isif
condition="${paymentMethodType.value.equals(dw.order.PaymentInstrument.METHOD_GIFT_
CERTIFICATE)}"><iscontinue/></isif>
```

Paste the following:

```
<iscomment>Start of Afterpay</iscomment>
<iscomment>Ignore the payment method AFTERPAY_PBI if Afterpay payment method is
disabled in configuration</iscomment>
<isif condition="${(paymentMethodType.value.equals('AFTERPAY') ||
paymentMethodType.value.equals('CLEARPAY')) && !afterpayApplicable}">
    <iscontinue/>
</isif>
<isif condition="${paymentMethodType.value.equals('CASHAPPPAY') &&
!cashAppEnabled}">
    <iscontinue/>
</isif>

<isif condition="${(paymentMethodType.value.equals('AFTERPAY') ||
paymentMethodType.value.equals('CLEARPAY'))}">
    <isset name="afterpayPayment" value="${paymentMethodType.value}" scope="page"/>
</isif>
<isif condition="${paymentMethodType.value.equals('CASHAPPPAY')}" >
    <isset name="cashAppPay" value="${paymentMethodType.value}" scope="page"/>
</isif>

<iscomment>End of Afterpay</iscomment>
```

Find and comment out the following code block:

```
<div class="form-row label-inline">
    <isset name="radioID" value="${paymentMethodType.value}" scope="page"/>
    <div class="field-wrapper">
        <input id="is-${radioID}" type="radio" class="input-radio"
name="${pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlName}
" value="${paymentMethodType.htmlValue}" <isif condition="${paymentMethodType.value
==
pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlValue}">check
ed="checked"</isif> />
    </div>
    <label for="is-${radioID}"><isprint
value="${Resource.msg(paymentMethodType.label, 'forms', null)}"/></label>
</div>
```

And replace with the following:

```
<div class="form-row label-inline">
  <isset name="radioID" value="${paymentMethodType.value}" scope="page"/>
  <div class="field-wrapper">
    <input
      id="is-${radioID}"
      type="radio"
      class="input-radio"

name="${pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlName}"
"
      value="${paymentMethodType.htmlValue}"
      <isset condition="${paymentMethodType.value ==
pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlValue}">
        checked="checked"
      </isset>
    />
  </div>
  <label for="is-${radioID}">
    <isset name="paymentObject" value="${paymentMethodType.object}"
scope="page"/>
    <isset condition="${radioID == 'AFTERPAY' || radioID == 'CLEARPAY'}">
      <isset condition="${countrycode == 'US'}">
        <square-placement
          data-mpid="${mpid}"
          data-type="logo"
          data-page-type="checkout"
          data-currency="${session.getCurrency().getCurrencyCode()}"
        ></square-placement>
      </isset>
      <img class="payment-image ${radioID.toLowerCase()}-image
${apBrand}-checkout-logo" alt="${Resource.msg('billing.logo', apBrand, null)}"/>
    </isset>
    </isset>
    <isset condition="${empty(paymentObject.image)}">
      <isprint
value="${Resource.msg(paymentMethodType.label, 'forms', null)}"/>
    </isset>
    
    </isset>
  </isset>
</label>
```

```
</div>
```

Locate the following block of code:

```
<iscomment>  
Custom processor  
-----  
</iscomment>
```

And paste the following below that code block, as seen in the screenshot:

```
<iscomment>  
    Afterpay Pay Over Time  
    -----  
</iscomment>  
<iscomment>Start of Afterpay</iscomment>  
<isif condition="${afterpayApplicable && !empty(afterpayPayment)}">  
    <div class="payment-method <isif condition="${!empty(pdikt.selectedPaymentID)  
&& pdikt.selectedPaymentID == afterpayPayment}">payment-method-expanded</isif>"  
    data-method="${afterpayPayment}">  
        <isafterpaymessage  
            classname="checkout-afterpay-message"  
            mpid="${mpid}"  
            totalprice="${afterpayOrdertotal}"  
        />  
    </div>  
    <isif condition="${afterpayExpressCheckoutEnabled &&  
isExpressCheckoutFinalize}">  
        <div class="afterpay-paymethod-widget-div">  
            <div id="afterpay-widget-container"></div>  
        </div>  
        <iscomment>  
            Disable the Continue to Place Order button during Afterpay Express Checkout  
            if Afterpay is selected as the payment option. The Afterpay button will be shown  
            instead.  
        </iscomment>  
        <isscript>  
            var placeOrderButtonClass = 'afterpay-placeorder-button';  
            if(countrycode === 'GB'){  
                placeOrderButtonClass = 'clearpay-placeorder-button';  
            }  
        </isscript>
```

```

<script type="text/javascript">
/**
 * If the Afterpay placeorder button is on screen, hide the original button
 */
function hidePlaceOrderButtonIfNecessary() {
  let element =
document.querySelector('.afterpay-placeorder-button')?document.querySelector('.afterpay-placeorder-button'):document.querySelector('.clearpay-placeorder-button');
  let originalContinueButtonElement = document.querySelector("#dwfrm_billing > div.form-row-button");
  if (!element || !originalContinueButtonElement) {
    return;
  }
  // Check if Afterpay radio button is selected
  var afterpayRadioButton =
document.querySelector('#is-AFTERPAY')?document.querySelector('#is-AFTERPAY'):document.querySelector('#is-CLEARPAY') ;
  if (afterpayRadioButton && afterpayRadioButton.checked == true) {
    // hide existing "CONTINUE TO PLACE ORDER" button

originalContinueButtonElement.classList.add("afterpay-hidden");
  } else {
    // unhide

originalContinueButtonElement.classList.remove("afterpay-hidden");
  }
}
document.addEventListener("DOMContentLoaded", function() {
  var paymentMethodOptions =
document.querySelector('.payment-method-options');
  if (paymentMethodOptions) {

paymentMethodOptions.addEventListener('change', function() {
hidePlaceOrderButtonIfNecessary()
  });
}
hidePlaceOrderButtonIfNecessary();
let elem = document.getElementById('afterpay-express-placeorder-button');
elem.addEventListener('click', function() {
  let finalizeUrl =
document.getElementById('afterpay-express-url-placeorder').value;

window.location.href=finalizeUrl+'?checksum='+afterpayWidget.paymentScheduleChecksum;
});
});
</script>

```

```

<div class="form-row form-row-button">
<input type="button" id="afterpay-express-placeorder-button"
class="{placeOrderButtonClass} afterpay-widget-hideuntilready">
&nbsp;
</input>
</div>
</isif>
</isif>

<iscomment>End of Afterpay</iscomment>

```

And for Cash App Pay

```

<iscomment>
    Cash App Pay
    -----
</iscomment>
<isif condition="{cashAppEnabled && !empty(cashAppPay)}">
    <div class="payment-method <isif
condition="{!empty(pdikt.selectedPaymentID) && pdikt.selectedPaymentID ==
cashAppPay}">payment-method-expanded</isif>" data-method="{cashAppPay}">
        <div class="cashapppay-checkout-message checkout-afterpay-message">
            ${Resource.msg('cashapppay.checkout.message', apBrand, null)}
        </div>
    </div>
</isif>

```

10. File:

app_storefront_core/cartridge/templates/default/product/components/pricing.isml

Locate this code:

```

<isinclude template="product/components/standardprice"/>

```

And paste the code below:

```

<iscomment>Start of Afterpay</iscomment>
<isinclude template="util/modules">
<iscomment>End of Afterpay</iscomment>

```

At the very bottom of the file, right before the last <isif>, paste the following:

```
<isscript>
  var priceForAp;

  if (pdict.ProductSetStandardPrice || pdict.ProductSetSalesPrice) {
    priceForAp = pdict.ProductSetSalesPrice < pdict.ProductSetStandardPrice ?
pdict.ProductSetSalesPrice : pdict.ProductSetStandardPrice;
  } else {
    priceForAp = pdict.Product.priceModel.isPriceRange() > 0 ?
pdict.Product.priceModel.minPrice : (SalesPrice ? SalesPrice : StandardPrice);
  }

  var sitePreferences =
require('*/cartridge/scripts/util/afterpayUtilities').sitePreferencesUtilities;
  var afterpayPDPEnable = sitePreferences.isDisplayPdpInfo();
</isscript>
<isif condition="${afterpayPDPEnable}">
<isinclude url="${URLUtils.url('Afterpay-RenderMessage',
  'totalprice', priceForAp,
  'classname', 'pdp-afterpay-message',
  'productID', Product.ID,
  )}"
/></isif>
```

11. File:

**app_storefront_core/cartridge/templates/default/product/producttile.is
ml**

At the top of the file, right after the </isif>, paste the following:

```
<iscomment>Start of Afterpay</iscomment>
<isinclude template="util/modules"></isinclude>
<iscomment>End of Afterpay</iscomment>
```

Around line 244 (after the above changes), comment out the following code:

```
prices.push(price);
if (extraPrice) {prices.push(extraPrice);}
```

Replace it by pasting in the following::

```
// Start of Afterpay
```



```

if (price && price.value) {prices.push(price);}
if (extraPrice && extraPrice.value) {prices.push(extraPrice);}
// End of Afterpay

```

Around line 254 (after the above changes), right before the line:

```
<iscomment>Promotion</iscomment>
```

Paste in the following as shown:

```

<iscomment>Start of Afterpay</iscomment>
<iscomment>Afterpay message</iscomment>
<isscript>
    var AfterpayUtilities = require('*/cartridge/scripts/util/afterpayUtilities');
    var BrandUtilities = AfterpayUtilities.brandUtilities;
    var priceForAp = pdict.CurrentHttpParameterMap.pricerange.stringValue ==
'true'? pdict.CurrentHttpParameterMap.minprice : prices[prices.length-1].value;

    BrandUtilities.initBrand(pdict.CurrentRequest.locale);
    var sitePreferences = AfterpayUtilities.sitePreferencesUtilities;
    var afterpayPLPEnable = sitePreferences.isDisplayPlpInfo();
</isscript>
<isif condition="${afterpayPLPEnable}">
<isinclude url="${URLUtils.url('Afterpay-RenderMessage',
    'totalprice', priceForAp,
    'productID', Product.ID,
    'classname', 'plp-afterpay-message',
    )}"
    />
</isif><iscomment>End of Afterpay</iscomment>

```

12. File:

app_storefront_core/cartridge/templates/default/checkout/cart/cart.isml

Near the top of the file, After

```
<isinclude template="util/reporting/ReportBasket.isml" />
```

paste the following as shown:

```

<iscomment>Start of Afterpay Express</iscomment>
<isif
condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('enableAfterpay')}
    &&
dw.system.Site.getCurrent().getCustomPreferenceValue('apEnableExpressCheckout')}">
    <isafterpayexpresscheckout basket="${pdict.Basket}">
        <iscomment>
            If the delivery option is changed, do ajax call to server to check whether
all items in
            cart are being picked up in-store. Call Afterpay Express widget with
"pickup" setting.
        </iscomment>
        <script type="text/javascript">
            document.addEventListener("DOMContentLoaded", function() {
                initAfterpay({pickupflag: $('#afterpay-express-storepickup').val() ===
"true"});
                initializeDeliveryOptionChangeListener();
            });
        </script>
    </isif>
<iscomment>End of Afterpay Express</iscomment>

```

Under the

```

<iscomment>show checkout and continue shopping buttons above cart</iscomment>

```

Paste following :

```

<isif condition="${pdict.Basket.totalGrossPrice.available}">
<isset name="orderTotalValue" value="${pdict.Basket.totalGrossPrice}"
scope="page"/>
    <iselse/>
    <isset name="orderTotalValue"
value="${pdict.Basket.getAdjustedMerchandizeTotalPrice(true).add(pdict.Basket.giftC
ertificateTotalPrice)}" scope="page"/>
    </isif>

```

Locate the following <div>:

```

<div class="cart-actions">

```

Right above it, paste the following:

```

<iscomment>Start of Afterpay</iscomment>
<isscript>
    var BrandUtilities =
require('*/cartridge/scripts/util/afterpayUtilities').brandUtilities;
    BrandUtilities.initBrand(pdikt.CurrentRequest.locale);
</isscript>

<div class="learn-more">
    <isinclude url="${URLUtils.url('Afterpay-RenderMessage',
        'totalprice', orderTotalValue,
        'classname', 'cart-afterpay-message'
        )}"
    />
</div>
<iscomment>End of Afterpay</iscomment>

```

Near line 910, under the <button> element, paste the following code:

```

<iscomment>Start of Afterpay Express</iscomment>
<isif condition="${afterpayExpressCheckoutEnabled &&
!disableAfterpayPaymentMethod}">
    <isscript>
        var AfterpayUtilities =
require('*/cartridge/scripts/util/afterpayUtilities');
        var BrandUtilities = AfterpayUtilities.brandUtilities;
        var countrycode = BrandUtilities.getCountryCode();
        var checkoutButtonClass = 'afterpay-checkout-button';
        if(countrycode === 'GB'){
            checkoutButtonClass = 'clearpay-checkout-button';
        }
    </isscript>
    <div>
        <isif condition="${isExpressCheckoutFinalize}">
            <a href="${URLUtils.url('AfterpayExpress-ContinueFinalize')}">
                <input type="button" class="button-fancy-large ${checkoutButtonClass}"
id="afterpay-continue-button" />
            </a>
            <iselse/>
                <input type="button" class="button-fancy-large ${checkoutButtonClass}"
id="afterpay-express-button" data-afterpay-entry-point="cart" />
            </isif>
        </div>
    </isif>

```

```
<iscomment>End of Afterpay Express</iscomment>
```

13. File: **app_storefront_core/cartridge/templates/default/util/modules.isml**

At the very bottom of the file, paste the following:

```
<iscomment>Start of Afterpay</iscomment>
<iscomment>
Render the Afterpay modules
</iscomment>
<isinclude template="util/modulesafterpay" />
<iscomment>End of Afterpay</iscomment>
```

14. File:

app_storefront_core/cartridge/templates/default/checkout/summary/summary.isml

Starting on line 221, comment out the following button:

```
<button class="button-fancy-large" type="submit" name="submit"
value="${Resource.msg('global.submitorder','locale',null)}">
    ${Resource.msg('global.submitorder','locale',null)}
</button>
```

Right underneath, paste the following code. This snippet contains the commented out button as well for clarity.

```
<!-- <button class="button-fancy-large" type="submit" name="submit"
value="${Resource.msg('global.submitorder','locale',null)}">
    ${Resource.msg('global.submitorder','locale',null)}
</button> -->
<iscomment>
    Removed for Afterpay
    <button class="button-fancy-large" type="submit" name="submit"
value="${Resource.msg('global.submitorder','locale',null)}">
        ${Resource.msg('global.submitorder','locale',null)}
    </button>
</iscomment>

<iscomment>Start of Afterpay</iscomment>
```

```

<isafterpaywidget />
<isif condition="${afterpayExpressCheckoutEnabled && isExpressCheckoutFinalize}">
<script type="text/javascript">
    document.addEventListener("DOMContentLoaded", function() {
        createAfterpayWidget();
        let elem =
document.getElementById('afterpay-express-placeorder-button');
        elem.addEventListener('click', function() {
            let finalizeUrl =
document.getElementById('afterpay-express-url-placeorder').value;

window.location.href=finalizeUrl+'?checksum='+afterpayWidget.paymentScheduleChecksum;
        });
    });
</script>
<isscript>
    var brandUtilities =
require("*/cartridge/scripts/util/afterpayUtilities.js").brandUtilities;
    var countrycode = brandUtilities.getCountryCode();
    var placeOrderButtonClass = 'afterpay-placeorder-button';
    if(countrycode === 'GB'){
        placeOrderButtonClass = 'clearpay-placeorder-button';
    }
</isscript>
<isif condition="${afterpayExpressCheckoutEnabled && isExpressCheckoutFinalize &&
afterpayAmount.value > 0}">
    <div id="afterpay-widget-container"></div>
</isif>
<div id="afterpay-placeorder" class="afterpay-widget-hideuntilready
afterpay-summary-button-container">
    <input type="button" id="afterpay-express-placeorder-button"
class="${placeOrderButtonClass} afterpay-summary">&nbsp;
    </input>
</div>

<elseif condition="${isCashApp}">
    <iscashappwidget />
<elseif/>
    <button class="button-fancy-large" type="submit" name="submit"
value="${Resource.msg('global.submitorder','locale',null)}">
        ${Resource.msg('global.submitorder','locale',null)}
    </button>
</isif>
<iscomment>End of Afterpay</iscomment>

```

15. `app_storefront_core/cartridge/templates/default/product/productcontent.isml`

Starting on line 222 it looks like SiteGenesis has a slight bug, which we will fix. The old line was:

```
<input type="hidden" name="cgid" id="cgid"
value="{pdict.CurrentHttpParameterMap.cgid.value}" />
```

Let's go ahead and add the \$ in the value attribute:

```
<input type="hidden" name="cgid" id="cgid"
value="$ {pdict.CurrentHttpParameterMap.cgid.value}" />
```

Right below this change, paste the code below:

```
<iscomment>Start of Afterpay</iscomment>
<isafterpayexpresscheckout pagetype="product_detail"/>
<isif condition="$ {afterpayExpressCheckoutEnabled &&
afterpayExpressCheckoutPdpEnabled}">
<script type="text/javascript">
    var continueFinalize = function() {
        var finurl = $('#afterpayurl-continuefinalize').val() +
"?cartAction=add&pid=" + $('#pid').val() + "&Quantity=" + $('#Quantity').val();
        window.location.href = finurl;
    }
    $(document).ajaxComplete(function() {
        if (document.querySelector("#afterpay-express-link-button")) {
            initAfterpay({pickupflag: false, target: "#afterpay-express-link-button",
productIdSelector: "#pid", productQuantitySelector: "#Quantity"});
        }
    });
    $(document).ready(function() {
        if (document.querySelector("#afterpay-express-link-button")) {
            initAfterpay({pickupflag: false, target:
"#afterpay-express-link-button", productIdSelector: "#pid",
productQuantitySelector: "#Quantity"});
        }
    });
</script>
</isif>
<iscomment>End of Afterpay</iscomment>
```

Right after line 245 of the change after

```
<button id="add-to-cart" type="submit" title="${buttonTitle}"
value="${buttonTitle}" class="button-fancy-large
add-to-cart">${buttonTitle}</button>
```

paste the following:

```
<iscomment>Start of Afterpay</iscomment>
<isif condition="${afterpayExpressCheckoutEnabled &&
afterpayExpressCheckoutPdpEnabled && !disableAfterpayPaymentMethod}">
<iscomment>
When calling initAfterpay, we pass in the selectors for the pid and Quantity input
fields for the item.
When Afterpay Express popup starts, it will grab the values using those selectors,
and pass those in as input to CreateToken.
The AfterpayExpress-CreateToken controller will add the product into the cart by
itself.
</iscomment>
<isscript>
var brandUtilities =
require("*/cartridge/scripts/util/afterpayUtilities.js").brandUtilities;
var countryCode = brandUtilities.getCountryCode();
var payNowButtonClass = 'afterpay-paynow-button';
if(countrycode === 'GB'){
    payNowButtonClass = 'clearpay-paynow-button';
}
</isscript>
<div class="afterpay-paynow-pdp">
<isif condition="${isExpressCheckoutFinalize}">
<input id="afterpay-express-continuefinalize-button" type="button"
class="${payNowButtonClass}" onclick="continueFinalize();return false;"></input>
<iselse/>
<input id="afterpay-express-link-button" type="button"
class="${payNowButtonClass}"></input>
</isif>
</div>
</isif>
<iscomment>End of Afterpay</iscomment>
```

16. File: **app_storefront_core/cartridge/js/pages/checkout/billing.js**

In the \$addCoupon.on() event handler, change the line:

```
if (data.success && data.baskettotal === 0) {
```

To:

```
if (data.success) {
```

Optionally, paste the following comments to indicate the reason for the change.

```
// Afterpay - fix for apparent bug since page should reload if coupon applied regardless  
// of baskettotal. Otherwise  
// order totals on page are not updated and no indication is given whether coupon has  
// been applied
```

As the comment indicates, SiteGenesis has a small bug on the billing page. If a coupon is applied, the order totals are not updated and there's no indication on the page whether the coupon was applied or not. The data is only updated if the page is refreshed.

17. File: **app_storefront_core/cartridge/js/app.js**

Near the bottom of the initializeEvents() function, paste the following:

```
// Start of Afterpay  
$(document).on('click', '.afterpay-link', function (e) {  
    e.preventDefault();  
  
    dialog.open({  
        url: $(e.target).attr('href')  
    });  
});  
// End of Afterpay
```


18. File: **app_storefront_core/cartridge/js/countries.js**

```
'use strict';

exports.init = function init () {
    $('.country-selector .current-country').on('click', function () {
        $('.country-selector .selector').toggleClass('active');
        $(this).toggleClass('selector-active');
    });
    // set currency first before reload
    $('.country-selector .selector .locale').on('click', function (e) {
        e.preventDefault();
        var url = this.href;
        var currency = this.getAttribute('data-currency');
        var locale = this.getAttribute('data-locale');
        $.ajax({
            dataType: 'json',
            url: Urls.setSessionCurrency,
            data: {
                format: 'ajax',
                currencyMnemonic: currency,
                locale: locale
            }
        })
        .done(function (response) {
            if (!response.success) {
                throw new Error('Unable to set currency');
            }
            window.location.href = url;
        });
    });
};
```

19. File: **app_storefront_core/cartridge/js/bonus-products-view.js**

Near the top of the file, paste the following:

```
function createOrder() {
    var url = util.ajaxUrl(Urls.createAfterpayOrder);

    ajax.getJson({
        url: url,
        callback: function (data) {
```

```

        if (data) {
            checkAuthorize(data);
        }
    }
});
}

function checkAuthorize(data) {
    if (window.AfterPay) {
        AfterPay.init();
        AfterPay.display({token: data.token});
    } else {
        setTimeout(checkAuthorize.bind(null, data), 3000);
    }
}

function initSubmitOrderEvent() {
    var $form = $('.checkout-billing');

    var $paymentMethods = $form.find('input[name$="_selectedPaymentMethodID"]');
    var $btnContinue = $form.find('button[name$="_billing_save"]');

    $btnContinue.on('click', function(e) {
        var methods = ['AFTERPAY', 'CLEARPAY'];
        if (methods.indexOf($paymentMethods.filter(':checked').val()) !== -1) {
            e.preventDefault();
            createOrder();
            return false;
        }
    });
    var $declined = $('.ap-declined'),
        $form = $('.submit-order');

    if ($declined.length > 0) {
        $form.on('submit', function (e) {
            e.preventDefault();
            window.location = window.Urls.billing;
        });
    } else {
        $form.off('submit').on('submit', function (e) {
            return true;
        });
    }
}

exports.init = function () {
    initSubmitOrderEvent();
}

```

```
};

module.exports.initSubmitOrderEvent = function () {
  initSubmitOrderEvent();
};
```

20. File: **app_storefront_core/cartridge/scss/default/style.scss**

At the end of the file, paste the following as shown in the screenshot:

```
// Start of Afterpay
@import "afterpay";
// End of Afterpay
```