

1. USUWANIE ETYKIETY „PROMOCJA” PRZY PRODUKCIE W KATALOGU PRODUKTÓW

Chcąc dodać produkt promocyjny do naszego sklepu, od razu w katalogu produktów, przy produkcie, który ma cenę promocyjną, pojawia się ten magiczny prostokąt z napisem „promocja „. Chyba nie tylko ja uważam że nie wygląda on zbyt ładnie. Są dwa proste sposoby na to żeby się go pozbyć. Jednym z nich jest opcja instalacji wtyczki, która pozwoli nam na zmianę wyglądu etykiety. Możliwość zmiany układu, czcionki, koloru czy położenia dostępna jest we wtyczce *Product Labels For Woocommerce (Sale Badges)* // jest to jeden z przykładów, ale osobiście miałem okazję ją testować więc serdecznie polecam. Jednak jesteśmy tu po to żeby poznać skrypt, który całkowicie pozwoli nam usunąć etykietę promocji, bo znajdą się osoby, którym zwyczajnie jest to nie potrzebne.

Skrypt, który należy umieścić:

```
add_filter('woocommerce_sale_flash', 'lw_hide_sale_flash');  
  
function lw_hide_sale_flash()  
{  
    return false;  
}
```

Skrypt należy umieścić w pliku functions.php



5. PRZELICZANIE WALUT W KOSZYKU Z PLN ZA POMOCĄ NBP API

Wszystkie zakupy jakich dokonuje, realizuje przez Internet, a często zdarza mi się płacić w obcej walucie (nie oszukujmy się, aktualnie trzymanie złotych czy to na koncie czy w przysłowiowej skarpecie jest w zupełności nieopłacalne). Ale praktycznie za każdym razem muszę sprawdzać za pomocą Google, aktualny kurs danej waluty. Dlatego wpadłem na pomysł że fajnie by było wyjść naprzeciw potencjalnemu klientowi i ułatwić mu to, oferując już gotowe rozwiązanie w koszyku w kwestii przeliczania waluty i to dodatkowo w czasie rzeczywistym. Więc zacząłem przemierzać Internet w poszukiwaniu rozwiązania – i znalazłem. Genialny kod, który pozwala wygenerować w koszyku automatyczny przelicznik walut z polskiego złotego na euro, dolary i funty. Oczywiście jeśli chcieli byśmy aby znalazła się w naszym koszyku inna waluta, jak najbardziej możemy ją dodać, ponieważ do funkcjonowania całego kodu posłuży nam [API z Narodowego Banku Polskiego](#).

Podsumowanie koszyka	
Kwota	794.00zł
Łącznie	794.00zł
Suma w EURO	166.75 EURO
Suma w USD	160.48 USD
Suma w GBP	148.51 GBP

Przejdź do płatności

Właśnie taki efekt uzyskamy jak dodamy do pliku `function.php` poniższy kod.

```
$waluty = array (
    'EURO' => 'eur',
    'USD' => 'usd',
    'GBP' => 'gbp'
);

add_action('woocommerce_cart_totals_after_order_total',
function() use ( $waluty ) {
    przeliczenie_waluty( $waluty );
}
);

function przeliczenie_waluty( $waluty ) {
    $amount = WC()->cart->total;
    echo '<table>';

    foreach($waluty as $key => $value) {

        $dane = file_get_contents('http://api.nbp.pl/api/exchangerates/rates/a/'.
        $value .'/?format=json');

        $json = json_decode($dane);

        $mid_kurs = $json->rates[0]->mid;

        $price = $amount / $mid_kurs;

        echo
        '<tr> <th>Suma w ' . $key. '</th> <td><strong><span>' . round($price, 2) .
        '</span><bdi>' . $key. '</bdi></strong></td> </tr>';

    }

    echo '</table>';
}
}
```

6. AUTOMATYCZNA ZMIANA STATUSU NA „ZREALIZOWANE” DLA PRODUKTÓW WIRTUALNYCH

Bardzo fajne rozwiązanie pozwalające na automatyczną zmianę statusu zamówienia dla produktów wirtualnych, które klient musi pobrać po zakupie. Jest to istotny element, ponieważ kiedy status zamówienia pozostanie „ w realizacji „, klient nie otrzyma maila z produktem, który zakupił i musimy zmieniać status ręcznie. Są oczywiście wtyczki, która pozwalają na automatyczną zmianę statusu ([kliknij tutaj](#)), ale my wykorzystamy kod, który umieścimy w pliku `functions.php`.

```
add_action( 'woocommerce_thankyou',  
'custom_woocommerce_auto_complete_order' );  
function custom_woocommerce_auto_complete_order( $order_id ) {  
    global $woocommerce;  
    if ( !$order_id )  
        return;  
    $order = new WC_Order( $order_id );  
    $order->update_status( 'completed' );  
}
```

7. USUŃ NIEPOTRZEBNE POLA KASY WEDLE WŁASNEGO UZNANIA

W przypadku sprzedaży produktów wirtualnych, niektóre pola kasy takie jak adres, miasto czy kod pocztowy są nam niepotrzebne. Fajnie było by się ich pozbyć i zostawić tylko te, które są wymagane w momencie składania zamówienia. Klasycznie poniższy kod umieszczamy w pliku `functions.php`.

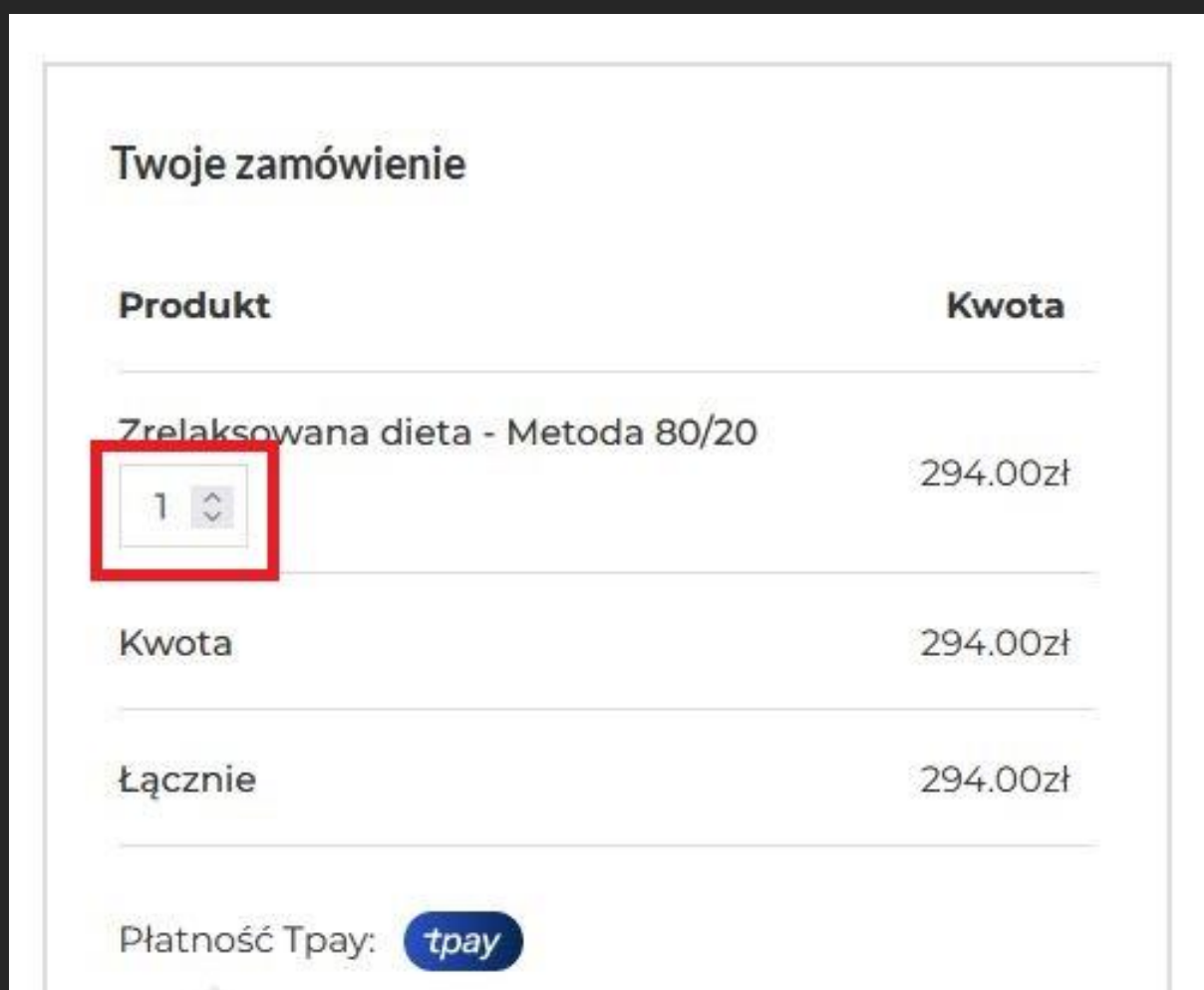
Usuniemy tutaj pole „ `Miasto – [‘billing_city’]` „ – w przypadku chęci usunięcia innych pól, najedź na interesujące Cię pole i naciśnij **prawy przycisk** myszki a następnie **Zbadaj**, przy `input id`, znajdziesz nazwę.

```
add_filter( 'woocommerce_checkout_fields', 'quadlayers_remove_checkout_fields' );  
function quadlayers_remove_checkout_fields( $fields ) {  
    unset($fields['billing']['billing_city']);  
    return $fields;  
}
```

17. ZMIENIĆ ILOŚĆ PRODUKTÓW NA STRONIE KASY

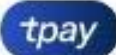
W tej części zakodujemy sobie świetny dodatek w obszarze kasy. Mianowicie dodamy sobie opcję ustawienia ilość danego produktu już na etapie finalizacji zamówienia. Jest to przydane rozwiązanie w przypadku, kiedy sprzedajemy produkty fizyczne. Założmy, że udzielamy rabatu w zależności od ilości produktów tego samego typu. Im więcej klient kupi tym większy rabat udzieli mu się w koszyku. Motywuje to klienta do zwiększenia ilości dodatkowo już na etapie finalizacji w kasie.

Ważne! To rozwiązanie nie działa w przypadku, kiedy ustawimy możliwość zakupu tylko jednego produktu tego samego typu.



Twoje zamówienie

Produkt	Kwota
Zrelaksowana dieta - Metoda 80/20	294.00zł
1	
Kwota	294.00zł
Łącznie	294.00zł

Płatność Tpay: 

oczywiście poniższy kod wklejamy do pliku functions.php:

```
add_filter( 'woocommerce_checkout_cart_item_quantity',
checkout_item_quantity_input', 9999, 3 );

function checkout_item_quantity_input( $product_quantity, $cart_item,
$cart_item_key ) {

$product = apply_filters( 'woocommerce_cart_item_product', $cart_item['data'],
$cart_item, $cart_item_key );

$product_id = apply_filters( 'woocommerce_cart_item_product_id',
$cart_item['product_id'], $cart_item, $cart_item_key );

if ( ! $product->is_sold_individually() ) {

$product_quantity = woocommerce_quantity_input( array(

'input_name' => 'shipping_method_qty_' . $product_id,

'input_value' => $cart_item['quantity'],

'max_value' => $product->get_max_purchase_quantity(),

'min_value' => '0',

), $product, false );

$product_quantity .= '<input type="hidden" name="product_key_' . $product_id .
"" value="" . $cart_item_key . "">'; }

return $product_quantity; }

add_action( 'woocommerce_checkout_update_order_review',
'bbloomer_update_item_quantity_checkout' );

function bbloomer_update_item_quantity_checkout( $post_data ) {

parse_str( $post_data, $post_data_array );

$updated_qty = false;

foreach ( $post_data_array as $key => $value ) {

if ( substr( $key, 0, 20 ) === 'shipping_method_qty_' ) {

$id = substr( $key, 20 );

WC()->cart->set_quantity( $post_data_array['product_key_' . $id],
$post_data_array[$key], false );

$updated_qty = true; } }

if ( $updated_qty ) WC()->cart->calculate_totals(); }
```

39. WYŚWIETL WYMAGANE BŁĘDY NAD POLEM W FORMULARZU KASY

Jedną z interesujących poprawek ułatwień dostępu jest system powiadamiania o błędach na stronie kasy. Tak, błąd brakujących pól pojawia się na górze strony podczas próby złożenia zamówienia, ale po przewinięciu w dół w celu ponownego wypełnienia może być potrzebne przypomnienie, którego pola brakuje, bez konieczności przewijania wstecz w celu sprawdzenia błędu.

```
add_filter( 'woocommerce_form_field', 'checkout_fields_in_label_error', 10, 4 );  
  
function checkout_fields_in_label_error( $field, $key, $args, $value ) {  
    if ( strpos( $field, '</label>' ) !== false && $args['required'] ) {  
        $error = '<span class="error" style="display:none">';  
        $error .= sprintf( __( '%s is a required field.', 'woocommerce' ), $args['label'] );  
        $error .= '</span>';  
        $field = substr_replace( $field, $error, strpos( $field, '</label>' ), 0 );  
    }  
    return $field; }  
}
```

Woocommerce JS dodaje klasę CSS o nazwie „woocommerce-invalid-required-field” do wymaganego pola, które nie jest wypełnione.

Każde pole otrzyma tę klasę i wygeneruje błąd. Na szczęście nie potrzebujemy JS do pokazania tych ukrytych rozpiętości, możemy po prostu celować w klasę.

```
.woocommerce-checkout p.woocommerce-invalid-required-field span.error {  
    color: #e2401c;  
    display: block !important;  
    font-weight: bold; }  
}
```

Imię * Imię jest wymaganym polem. <input type="text"/>	Nazwisko * Nazwisko jest wymaganym polem. <input type="text"/>
--	--