

Live jQuery Cart – Instant Update of Shopping Cart Totals

- [Live jQuery Cart – Instant Update of Shopping Cart Totals](#)

Retail Cost	Today's Special
\$125.00	\$80.00
Current Savings: \$45.00	
Quantity: <input type="text" value="1"/>	
ADD TO CART	

Why a Live jQuery Cart?

Customers love a simple shopping experience... with that, a short join process that includes the totals up front, produces less junk form submits, thus better form conversion ratios. This is uniquely important when it comes to split testing a live cart. Because the live cart has very low junk submits, a micro change to the form (be it design, location of the form, wording, a second step, etc), can “easily” be identified as positive or negative in the ratios.

How does the Live jQuery Cart Work?

When the quantity field is updated, the jQuery script is called, which takes the value of quantity and multiplies it by the retail and for sale price values. The retail, today's price, savings totals and a hidden input price field get instantly updated by using the jQuery `replaceWith` function.

Breaking down the Live jQuery Cart

The Form

The form includes three price point displays, the retail cost, today's special and current savings. When the quantity input field is modified, the jQuery is called, adjusting the three totals.

```
<div class="cart">
<!-- default retail, cost and savings values -->
  <div class="price">
    <div class="retail">
      <p>Retail Cost</p>
      <span class="pRetail">$125.00</span>
    </div>
    <div class="today">
      <p>Today's Special</p>
      <span class="pToday">$80.00</span>
    </div>
    <br style="clear:both" />
  </div>
  <div class="savings">Current Savings: <span class="save">$45.00</span></div>
<!-- form -->
  <form action="" method="post">
    <input type="hidden" name="price" value="" class="input" />
    <p><label>Quantity: <input type="text" name="quantity" size="2" value="1" /></label></p>
    <input type="submit" class="addCart" />
  </form>
</div>
```

The CSS

The CSS contains nothing fancy. The only trick to it is the button, which is displayed through a background call so the transparency works.

```
/*
  Name: Live jQuery Cart
  URI: http://techNerdia.com/
  Author: tribalNerd
*/
.cart {
  margin:0 auto;
  width:430px;
}
.price {
  margin:15px auto;
  text-align:center;
  width:325px;
}
.retail {
  color:#cc0000;
  float:left;
  font-size:14px;
  font-weight:bold;
  margin:0 0 2px 0;
  text-align:center;
  width:43%;
}
.today {
  color:#7241a8;
  float:right;
  font-size:14px;
  font-weight:bold;
  margin:0 0 2px 0;
  text-align:center;
  width:43%;
}
.retail p, .today p { margin:0 0 2px 0; }
.pRetail, .pToday {
  font-size:32px;
  font-weight:bold;
  line-height:34px;
}
.pRetail {
  text-decoration:line-through;
}
.savings, .save {
  color:#337147;
  font-size:22px;
  font-weight:bold;
  text-align:center;
}
.cart form, .cart label {
  margin:0 auto;
  padding:0;
  width:427px;
}
.cart form p, .cart label { text-align:center; }
.addCart {
```

```

background:url('cartButton.png')no-repeat;
background-image:url('cartButton.png');
background-position:0 0;
background-repeat:no-repeat;
border:0;
display:block;
height:110px;
margin:0 auto;
padding:0;
text-indent:-9999px;
width:427px;
}

```

The jQuery Cart

The jquery-cart.js file is a fun one....

- The script starts off by detecting the key change within the quantity field, it then scrubs the form (to blank) if the quantity field starts with a zero, a letter or a period.
- After the input value is good, the script checks to see if the quantity value is blank, zero or set to one – for display purposes. If so, the default prices kicks in. Else, the quantity field is 2 or more and the math starts.
- The math takes the quantity value and multiplies it by the retail or price value (today's cost). The saved value is the difference between the retail cost and today's cost.
- Finally the output happens: When the quantity field updates, the the jQuery executes a [replaceWith function](#). This replaces the 4 html elements above (based on the class names), with the 4 new html lines from the jquery-cart.js file (below the /* output */ line). This is what makes the price values change on the fly.

```

$('input[name=quantity]').keyup( function() {
  var value = $(this).attr("value");
  while ( value.substring(0, 1) === '0' || value.length > 3 ) { /* no leading zero, max 3 chars */
    value = value.substring(1); /* return to nothing */
  }
  value = value.replace(/[^0-9\.\,]/g,""); /* numbers only */
  value = value.replace(/\./g,""); /* no periods */
  $(this).val(value);
  var real = '125.00'; /* retail price */
  var cost = '80.00'; /* for sale price */
  if ( value == "" || value == "0" || value == "1" ) { /* default */
    var retail = "";
    var saved = "";
    value = '1';
    retail = real;
    saved = real - cost;
  }else{
    var total_cost = "";
    var retail = "";
    var saved = "";
    total_cost = parseInt(value) * cost;
    retail = parseInt(value) * real;
    saved = retail - total_cost;
  }
}

```

```

    }
    var total = "";
    total = parseInt(value) * cost;
/* output */
    $(".pRetail").replaceWith('<span class="pRetail">' + retail + '</span>');
    $(".pRetail").formatCurrency();
    $(".pToday").replaceWith('<span class="pToday">' + total + '</span>');
    $(".pToday").formatCurrency();
    $(".save").replaceWith('<span class="save">' + saved + '</span>');
    $(".save").formatCurrency();
    $(".input").replaceWith('<input type="hidden" name="price" value="" + total + "" class="input" />');
    $(".input").formatCurrency();
});

```

Required Changes

Within the jquery-cart.js file, Lines 9 and 10 need to be updated. Line 9 is the retail price of the product. Line 10 is the for sale price.

```

var real = '125.00'; /* retail price */
var cost = '80.00'; /* for sale price */

```

The jquery-cart.js also contains 4 **replaceWith** lines, when the jQuery file is executed the replaceWith commands updates the HTML in the form.

The replaceWith lines from the jquery-cart.js file.

```

$(".pRetail").replaceWith('<span class="pRetail">' + retail + '</span>');
$(".pToday").replaceWith('<span class="pToday">' + total + '</span>');
$(".save").replaceWith('<span class="save">' + saved + '</span>');
$(".input").replaceWith('<input type="hidden" name="price" value="" + total + "" class="input" />');

```

The above jQuery lines are: line 28, 30, 32 and 34. Below are related the HTML lines from the form. The lines are: Line 6, Line 10, Line 14 and Line 17. The jQuery lines, update the HTML lines, and it does this by matching the class names.

Matching lines from the HTML Form.

```

<span class="pRetail">$125.00</span>
<span class="pToday">$80.00</span>
<span class="save">$45.00</span>
<input type="hidden" name="price" value="" class="input" />

```

In most cases, the replaceWith and related HTML lines will need to be updated to match your Websites shopping cart forms. Most shopping carts require extra hidden values to be passed in, like the quantity and many other possible values. Static fields, like store names go into the HTML form directly and would not need to be replaced.

The Currency

The rounding of the currency and currency used, is managed by [jQuery Format Currency Plugin](#) available at Google Code.

- [Download the jquery.formatCurrency-1.4.0.js file from Google Code.](#)
-

**[Follow techNerdia on Google+](#) | [Follow Chris on Google+](#)
[Subscribe to techNerdia's RSS Feed](#)**

Copyright 2012 © [techNerdia](#) - All Rights Reserved
<http://techNerdia.com/>