shoppingCart Documentation

Release 1.3.0

October 26, 2010

CONTENTS

CHAPTER

ONE

SHOPPINGCART

1.1 Introduction

1.1.1 Overview

shoppingCart is a open source cart developed using Python language to manage cart in Ecommerce applications.

Main Feature(s):

1. Product options support.
2. Multi discount support.
3. Multi tax support(specific to product as well as general taxes).
4. Multi currency support.
5. Tax Exclude and Include total.
6. Shipping method and charge.

1.1.2 Installation

To install it, make sure you have **Python 2.5 or greater** and **python-setuptools** module is installed. Then run this command from the command prompt:

python setup.py install

OR you can also install it using **easy_install** command:

easy_install shoppingCart

AS AN ALTERNATIVE, you can just copy the entire "shoppingCart" directory to Python's site-packages directory, which is located wherever your Python installation lives. Some places you might check are:

/usr/local/lib/python2.6/dist-packages (Unix, Python 2.6)

/usr/lib/python2.5/site-packages (Unix, Python 2.5)

C:\PYTHONsite-packages (Windows)

1.2 Objects

1.2.1 Cart Object

class shoppingCart.cart.**Cart**(*site=None*, *customer=None*)
Collection of CartItem objects.

add_item(*product*, *price=0.0*, *quantity=1*, *taxes=[]*, *options={}*)
To add or update product, price, quantity, taxes and options in CartItem object.

Parameters

- **product** – Unique id or name of Product object or instance of Product.
- **price** – Product price.
- **quantity** – Product quantity(default 1).
- **taxes** – Taxes of the product(default []).
- **options** – Options of the product(default {}).

update_item(*product*, *quantity*, *option_values=[]*)
To update CartItem object quantity.

Parameters

- **product** – Unique id or name of Product object or instance of Product.
- **quantity** – Updated quantity.
- **option_values** – Option values of the product(default []).

remove_item(*product*, *option_values=[]*)
To remove existing CartItem object related to product.

Parameters

- **product** – Unique id or name of Product object or instance of Product.
- **option_values** – Option values of the product(default []).

remove_items()
To remove all existing CartItem objects.

find_item(*product*, *option_values=[]*)
To find CartItem object related to product.

Parameters

- **product** – Unique id or name of Product object or instance of Product.
- **option_values** – Option values of the product(default []).

Returns CartItem object if cart item is exist else None.

get_items()

Returns List of CartItem objects.

add_discount(*amount*, *type='percentage'*)
To apply discount.

Parameters

- **amount** – Discount amount.

- **type** – Discount type like 'percentage' or 'amount'(default amount).

remove_discounts()
To remove all applied discounts.

get_discounts()

Returns List of applied discounts.

add_tax(*amount*, *type='percentage'*)
To apply taxes.

Parameters

- **tax** – Tax amount according to country region.
- **type** – Tax type like 'percentage' or 'fixed'(default percentage).

Returns True if tax is applied and tax is already exist then False.

remove_tax(*amount*, *type='percentage'*)
To remove existing tax.

Parameters

- **amount** – Tax amount according to country region.
- **type** – Tax type like 'percentage' or 'fixed'(defualt percentage).

remove_taxes()
To remove all applied taxes.

find_tax(*amount*, *type='percentage'*)
To find applied tax.

Parameters

- **amount** – Tax amount according to country region.
- **type** – Tax type like 'percentage' or 'fixed'(defualt percentage).

Returns True if tax is exist else False.

get_taxes()

Returns list of applied taxes.

sub_total()

Returns Total cart amount(without discount deduction).

total_discount()

Returns Total discount amount.

total_untaxed_amount()

Returns Untaxed amount after deducating discount amount.

total_tax()

Returns Total tax amount.

total()

Returns Total amount(*tax excluded* or *tax included*) by adding total untaxed amount, total tax and shipping charge.

count()

Returns Total quantity.

clear()
To clean `Cart` Object.

is_empty

Returns True if cart has an item else False.

is_discount_applied

Returns True if discount is applied else False.

has_taxes

Returns True if tax is applied else False.

shipping_charge

Returns Shipping charge.

1.2.2 CartItem Object

class shoppingCart.cart.**CartItem**(*cart*, *product*, *price*, *quantity*, *taxes*=[ ], *options={}*)
Collection of `Product` and it's quantity, taxes and options.

update_quantity(*quantity*)
To update existing quantity related to `Product` object.

Parameters

- **quantity** – Product quantity.

get_options()

Returns Dict of product's option.

get_taxes()

Returns List of applied taxes.

sub_total()

Returns Total amount by multiplying product price and quantity(without discount deduction).

discount_amount()

Returns Discount amount.

untaxed_amount()

Returns Untaxed amount(after deducating discount amount).

tax_amount()

Returns Tax amount.

total()

Returns Total amount(*tax excluded* or *tax included*) by adding untaxed and taxed amount.

has_options

Returns True if product has an option else False.

has_taxes

Returns True if product has tax else False.

price

Returns Price by multiplying currency rate.

CHAPTER

TWO

LINKS

pypi page: http://pypi.python.org/pypi/shoppingCart

repository page: http://bitbucket.org/dharmeshpatel/shoppingcart

document link: https://bitbucket.org/dharmeshpatel/shoppingcart/downloads/shoppingCart_user_guide.pdf

CHAPTER

THREE

LICENSE

This documentation is under the GFDL

The GNU Free Documentation License. See http://www.gnu.org/copyleft/fdl.html.

CHAPTER

FOUR

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

INDEX