Silverstripe E-commerce Customisation Chart

Once you have successfully installed your e-commerce application, it is time to start customising. This flow chart shows you what steps to take in what order (from easiest to hardest).

1. Can it be fixed by a setting in the Database?	yes	Update database fields using the CMS.	(1) http://www.mysite.com/admin/shop/ and check settings (2) create required pages (3) check fields for e-commerce pages (e.g. messages in Checkout Page)
2. Is it a wording change?	yes	Create your own language file and add your own translation.	(1) do a search for the text in your source code. (2) what is your default AND current user locale? (3) create new folder "z" with _config.php in it, add "lang" directory with "_manifest_exlude" file. (4) In the lang directory, add the required translation file (see cms/lang/ for examples). Also see: http://doc.silverstripe.org/framework/en/topics/i18n
3. Can it be fixed by changing the config files?	yes	Update mysite/_config/ecommerce.yaml and/or mysite/_config.php file	(1) go to http://www.mysite.com/dev/ecommerce/ecommercecheckconfiguration (2) check ecommerce/_config.php for ideas (as well as sub-module _config.php files) (3) search code for "function set_myvariable" to see if there are any methods available to configure your application.
4. Can it be fixed with CSS / JS	yes	Theme appropriate CSS file and adjust as needed or block a JS / CSS file and add your own requirement.	(1) Check the page source header to see what CSS files are used (2) try to "Theme" the most appropriate CSS file. If that does not work, use Requirements::block(oldFile) and Requirements::css(newFile). (3) Use !important in css definition if you have no other choice. For more information on theme-ing, visit: http://doc.silverstripe.org/framework/en/topics/theme-development.
5. Can it be fixed by adjusting the HTML?	yes	Theme appropriate template file (.ss) and change as needed.	(1) Check the source to see what HTML files are used (add SSViewer::set_source_file_comments(true); to your _config.php file. (2) "Theme" the most appropriate HTML template (.ss) file - use 'unthemed" file as starting point and check the relevant Controllers for available "Controls". Only theme what is needed. For more information on theme-ing, visit: http://doc.silverstripe.org/framework/en/topics/theme-development.
6. Is there a sub-module that can add the functionality you are looking for?	yes	Install sub-module.	(1) Visit http://www.silverstripe.org/modules (2) Visit http://www.ssmods.com/developers/all-silverstripe-modules/#filter_ecommerce (3) Install as per usual - making sure that the versions match.
7. Do you want to change the display / functionality of products?	yes	Extend the Product / ProductGroup class.	(1) Make a copy of Product(Group).php and call it MyProduct(Group). The class should read class Product(Group) extends Product(Group). (2) Delete any variables / methods that you are happy with and alter / update / add new variables and methods needed. (3) Add static $hide_ancestor = "Product(Group)" to hide the default Product(Group). (4) If you need to, you can also extend the relevant OrderItem (e.g. MyProduct_OrderItem extends Product_OrderItem)
8. Do you want to add a new type of product that does not suit Product or Product Variation?	yes	You can create your own "Buyable" class.	(1) Make sure you use the BuyableInterface (e.g. MyBuyable extends DataObject implements BuyableInterface), which will help you add the right "methods" (functions) to your custom Buyable Class. (2) Make sure to update ecommerce.yaml with the additional Buyable.
9. Do you want to add a discount / charge / tax other crazy thing that is dependent on the content of the order?	yes	Create a class that extends OrderModifier	(1) Look at the example modifier: https://github.com/sunnysideup/silverstripe-ecommerce_modifier_example to get some ideas. (2) Make sure to add the modifier in the mysite/_config/ecommerce.yaml file in the Order.modifiers variable. (3) If your OrderModifier includes any options (e.g. see /ecommerce_tax/code/model/GSTTaxModifierOptions.php) then add to StoreAdmin (see /mysite/_config/ecommerce.yaml)
10. Do you want to add a general note to your order that is not dependent on the items added to the order?	yes	Create an Order Status Log.	(1) Have a look at /ecommerce/code/model/process/ to see what OrderStatusLog classes are included by default. (2) Explore the e-commerce sub-modules for more examples (3) Add MyOrderStatusLog to /mysite/_config/ecommerce.yaml file in the OrderStatusLog.available_log_classes_array variable. (4) Also, OrderStatusLogs are not added automatically, so you will need to use an OrderStep to create them. Again, the core code will provide some good examples.
11. Do you want to change something in the Order Process? E.g. sending an extra e-mail, beaming the order to a third-party app or just shortening the process.	yes	Add / Edit / Remove any of the OrderStep	(1) Check /admin/shop (Order Steps), to see what you can add/edit/delete without coding. (2) Open /ecommerce/code/model/process/ in your code and look at the OrderStep classes created here. Find the one that is closes to what you want to do and create your own class: MyOrderStep extends OrderStep. Make sure to review the core methods. (3). Add MyOrderStep to /mysite/_config/ecommerce.yaml file in the OrderStep.order_steps_to_include variable.
12. Do you want a different payment method?	yes	Create your own payment class.	Strictly speaking, the Payment module is outside of the scope of e-commerce, but it is a requirement of course. There are plenty of Payment Class examples out there... Create your own class and add a reference to it in your Payment::set_supported_methods(...); call /mysite/_config.php.
13. Can you extend a class to achieve your result?	yes	Use the standard MyClass extends CoreClass and override any methods and variables you want changed.	See http://doc.silverstripe.org/framework/en/topics/datamodel for more information.
14. Can a class "decorator" / "extension" sort out your worries?	yes	Add a decorator / extension	Within the e-commerce code, we have added a number of "hooks", they often look like this: $this->extend or similar... you can use these "hooks" to change the actions and outputs of core methods. See: http://doc.silverstripe.org/framework/en/topics/datamodel for more information.
15. Can you create a custom class that replaces the core class?	yes	Create custom class to replace a core one.	Create your custom class and add the following to your mysite/_config.php file: Object::useCustomClass('CoreClass','MyClass');
16. Can you hack the core to make it work?	yes	Make a fork on Github or just start hacking away.	A fork on github is the preferred method here. If you are using SVN, you can look at Piston as a solution.
17. Contact the e-commerce developers to discuss options.			Project Home: http://code.google.com/p/silverstripe-ecommerce/ GIT Home: https://github.com/sunnysideup/silverstripe-ecommerce Demo Home: http://www.silverstripe-ecommerce.com