SuiteCommerce: *Building Web Stores Part 1*

SuiteTraining™

Course Description

Do you need to create a highly custom SuiteCommerce Advanced or SuiteCommerce SiteBuilder web site? This 3-day course shows you how to implement and extend prebuilt reference implementations for My Account, Cart, and Checkout that are all running on the same front-end web architecture utilizing Backbone, Underscore, Bootstrap, and jQuery. Take advantage of SuiteScript Server Pages technology to allow for a high degree of customization across your website.

This course is the first part of a two-course series. This course (Part 1) should be taken by those implementing SuiteCommerce SiteBuilder or SuiteCommerce Advanced websites. SuiteCommerce: Building Web Stores Part 2 should be taken only by those implementing SuiteCommerce Advanced. SuiteCommerce Advanced customers should take both courses.

Who Should Attend

- Software developers needing to create unique and highly customized websites using SuiteCommerce Advanced or SuiteCommerce SiteBuilder

Prerequisites

- Course participants should have a significant programming background
- JavaScript is used in the customization of web stores, both on the client (browser) and server. It is recommended that you have some exposure to JavaScript, including syntax relevant to objects and anonymous functions
- Exposure to the browser DOM and browser event model is also helpful, including use of CSS selectors
- Experience with browser-based JavaScript libraries such as Backbone, Underscore, and jQuery is a plus.

Hardware / Software Requirements

- Windows, Mac, or Linux device
- Chrome or Firefox browser. Install the Firebug add-on if using Firefox
- SuiteCloud IDE. Search for the topic "Installing SuiteCloud IDE" in the NetSuite Help Center for details.
- Install an additional Html/JavaScript editor. Some options are Sublime Text 2, Komodo Edit, and Notepad++. You can find these easily by searching on the web

Key Tasks

How do I:

- Customize and extend Reference Checkout?
- Customize and extend My Account?
- Build entirely custom SSP applications on the SuiteCommerce platform?
- Interact with the Commerce API?
- Interact with the SuiteScript API in a web store context?
- Build new modular components that run seamlessly inside of Reference Checkout or My Account?
- Implement my changes to web store code in a manner that supports maintainability, performance, and developer productivity?

Related Courses

- SuiteScript: Extend NetSuite with JavaScript
    - Some SuiteScript is covered in this Building Web Stores Part 1 course, but the SuiteScript course goes deeper into an array of SuiteScript topics with specific focus on automation inside of NetSuite

SuiteAnswers

Get answers to your support and training related questions:

- Find related self-paced training videos at the Learning Center
- Learn about the latest NetSuite release with New Feature Training

Live Training Webinars

- Go to suitetraining.com > Training Webinars to view the schedule and register for a free event and get practical tips

Course Objectives and Topics

This course takes you through the implementation of a SuiteCommerce web store with focus on extending Reference Checkout and My Account, including custom SSP application development. A variety of best practices are covered, including approaches on how to extend the reference implementations. SSP applications are built and extended in the back-end via SuiteScript Services, SuiteScript API, Commerce API, and SSP libraries. SSP applications are built and extended in the front-end using various open source JavaScript libraries that make up the reference implementation architecture.

Day 1 Agenda

Summarize Web Store Technologies: Define the NetSuite record structure; define the SuiteCloud platform technologies; compare SuiteCommerce products; define architecture of SuiteCommerce Reference Implementations; set up a SuiteCommerce Advanced or SuiteCommerce SiteBuilder web store

Build SSP Applications using SuiteScript API: Build a SuiteScript Server Page (SSP); incorporate the SuiteScript API into an SSP; debug and troubleshoot an SSP; define SSP application precedence

Build SSP Applications using Commerce API: Incorporate the Commerce API into an SSP; use SSP libraries in your SSP application, configure your SSP for custom field support

Incorporate SuiteScript Services into SSP: Gain an understanding of when to use a SuiteScript Service (SS); build and test SuiteScript Services; add support to SuiteScript Services for http GET, POST, PUT, and DELETE methods; organize service code inside SSP libraries; identify when you might need to incorporate Suitelets into your SSP application

Day 2 Agenda

Investigate Reference Implementations: Gain an understanding of server-side and client-side reference implementation architecture; inspect the SuiteCommerce global JavaScript variable, identifying relevant areas of data; change a front-end JavaScript file; change a front-end template file; effectively debug JavaScript and template files

Modify Reference Checkout: Configure the layout of checkout steps; identify how settings on the Web Site Setup record can be evaluated in your code; add new configuration data and use it to alter front-end behavior; identify use of Backbone. Events in front-end code, including how to identify pre-existing event triggers; modify a back-end model in such in such a way the impact of future upgrades are lessened

Modify My Account: Modify content in My Account using an alternative approach to precedence that lessens impact of future upgrades; modify front-end JavaScript; modify front-end template and macro files

Day 3 Agenda

Build Back-end of New Module: Create a new back-end model for a reference implementation; create a new SuiteScript Service for a reference implementation; update a reference implementation to incorporate new record types; enable the back-end portion of a new module to operate inside of My Account

Build Front-end of New Module: Update My Account application start up JavaScript to reflect new data from back-end; declare new module to a reference implementation; add module components to support Backbone Collection, Model, Router, and View; add templates and macros to support Backbone Views; configure back-end service layer to support form submittal

Extend Module: Incorporate client-side and server-side validation via Backbone. Validation; extend a back-end model through update and override of model methods; extend a back-end model via server-side event triggers; extend front-end code via event triggers

Translate Languages: Translate labels inside of templates, macros, views, models, and other front-end JavaScript files; translate item data

Design Your Web Store: Identify usage of Bootstrap in SuiteCommerce web stores; gain an understanding of LESS; identify the process of modifying site styles