Insite for Sitecore 3.3

Configuration Guide

A developer's guide to configuring Insite for Sitecore

Table of Contents

Chapter 1

Introduction

This document explains how you configure and work with Insite for Sitecore.

This document contains the following chapters:

- **Chapter 1 — Introduction**
  This chapter gives a summary of the content contained in each chapter.

- **Chapter 2 — Product Synchronization**
  This chapter describes how the product synchronization process works.

- **Chapter 3 — Insite API & Sitecore**
  This chapter explains how to access the Insite API from Sitecore components.

- **Chapter 4 — SES API Extensions**
  This chapter explains the SES API extensions provided with Insite for Sitecore.

- **Chapter 5 — Web Forms For Marketers**
  This chapter describes Insite for Sitecore's integration with Web Forms For Marketers.

- **Chapter 6 — DMS Integration**
  This chapter describes Insite for Sitecore's integration with DMS.

Chapter 2

Product Synchronization

Product synchronization is the process by which products managed by Insite are exposed as Sitecore items. Each Insite product has a corresponding Sitecore item. Product synchronization is the process that handles the creation of the Sitecore item, and it ensures the Sitecore item stays in sync with the corresponding Insite product.

2.1 Product Configuration Items

In Insite Connect, each product managed by Insite has a corresponding Sitecore item. A Product Configuration item controls how the Sitecore item should be created and populated.

Multiple Product Configuration items are support. This allows different types of products to be handled differently. For example, cameras may use one template while lenses use a different template. Or products for a specific webshop belong in a specific section of the content tree.

Configuration Rules are used to specify which Product Configuration item applies to an Insite product. Using Configuration Rules is optional.

2.1.1   Default Configuration Item

Insite Connect has a setting that identifies the default Product Configuration item. This is the item that is used if, after the Configuration Rules run, no Product Configuration item has been identified.

The default Product Configuration item is specified by the following field:

Item path	`/sitecore/system/Modules/Insite`
Field name	`Default Product Configuration`
Default value	`/sitecore/system/Modules/Insite/Product Configurations/Default Product Configuration`

2.1.2   Product Configuration Fields

The following fields are available on Product Configuration items.

Field name	Description
Enabled	If checked the configuration settings are used. This field can be used to disable syncing.
Product Template	The data template to use when creating new Sitecore items.
Product Repository Base	The Sitecore item under which product items are created.
Overwrite From External System	This field only applies to the item name. When a product item is created, it is possible for content authors to make changes to the item, including changing the item name.
	If this option is checked the item name will be overwritten using the value from the external system. If this option is not checked the item name will not be changed.
Item Name Property	The field whose value is used for the product item name.
Field Mapping	The field mapping item that determines how the fields on the product item are populated..
Structure Rule	This field determines the folder structure used to organize product items. The default value is "Structure By Product Id Rule" which results in product items being organized into folders based on the item id.

2.2 Field Mapping Items

A Product Configuration item uses field mappings to determine which Insite fields are exposed by the Sitecore product item. This section describes the different items that are used to configure these mappings.

2.2.1 Field Mapping Types

Insite exposes product data in multiple ways. Depending on the data needed, a specific API call is used. Field Mapping Type items are used to identify which API call is needed in order to read a specific value.

The following Field Mapping Types are supported:

- **Category** - The categories a product is assigned to
- **Custom Property** - A custom property on the product
- **Property** - A built-in property on the product

Field Mapping Types are defined under /sitecore/system/Modules/Insite/Field Mapping Types/Insite.

2.2.2 Field

A Field item represents a specific value on an Insite product. In order to be able to read a value from an Insite product, a Field must be defined. If a Field is not defined Sitecore has no way to know what API calls to make in order to read a value from the Insite product.

By default the following fields are available:

- Category
- ERPDescription
- ERPNumber
- LargeImage
- MediumImage
- Name
- ProductId
- ShortDescription
- SmallImage

Each Field item has the following fields:

- **Field Name** - For Property and Custom Property fields, the Field Name must match the property name in Insite.
- **Field Type** - The Field Mapping Type that identifies how Sitecore should read the field value from the Insite product.
- **Description** - A description of the Field.

Field items are defined under /sitecore/system/Modules/Insite/Fields.

2.2.3 Field Mapping

A Field Mapping item specifies how a Sitecore field should be populated.

Each Field Mapping item has the following fields:

- **Sitecore Field** - Name of the field on the Sitecore item to populate
- **External Field** - Field from the Insite product whose value is used to populate the field specified in the Sitecore Field field
- **Overwrite From External System** - The field on the Sitecore item will only be updated from Insite if this option is checked

Field Mapping items are children of a specific Field Mappings item.

2.2.4   Field Mappings

A Field Mappings item is a collection of individual Field Mapping items.

Each Field Mappings item has the following fields:

- **Use Mappings From** - This field allows you to inherit mappings from other Product Configuration items

Field Mappings items are defined under /sitecore/system/Modules/Insite/Field Mappings.

2.3 Pipelines

Insite uses NHibernate to map its domain model to a relational database. NHibernate supports interceptors, which are components that are triggered when certain events occur (such as when changes to the domain model are saved to the database).

Insite Connect includes a custom interceptor that triggers Sitecore pipelines that control the product synchronization process. This section describes these pipelines.

2.3.1   saveInsiteProduct

When a product is saved in Insite, this pipeline is triggered.

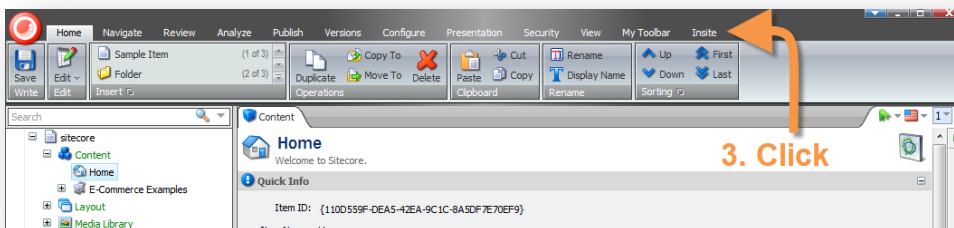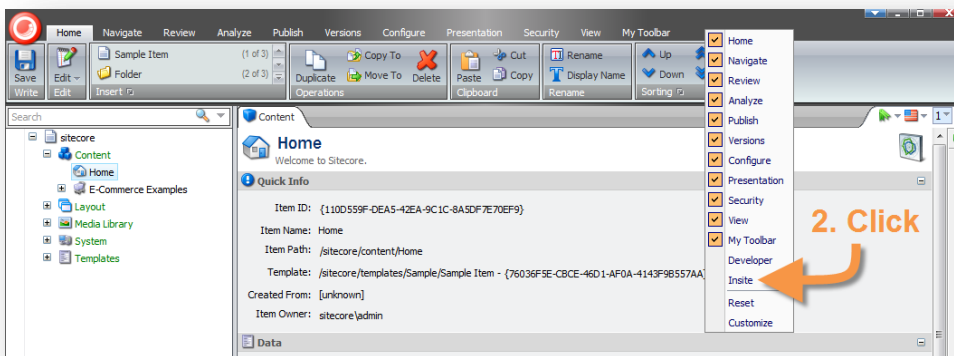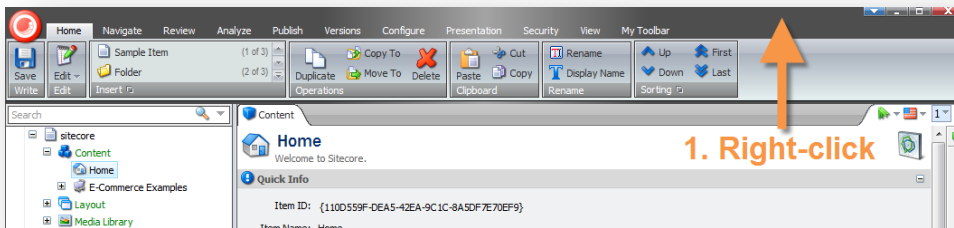### 2.3.2   deleteInsiteProduct

When a product is deleted from Insite, this pipeline is triggered.

2.4 Manual Product Sync

When products are created, updated and deleted from Insite, Sitecore pipelines are called to keep Sitecore in sync with Insite. Changes are pushed from Insite to Sitecore. This happens automatically.

But there are times when you might want to refresh a Sitecore item (in other words, you might want to pull content from Insite to Sitecore). This section describes how to initiate the product sync process from Sitecore.

2.4.1 Show Insite Ribbon

2.4.2 Sync All

The Sync All button performs a search for all Insite products and then runs the saveInsiteProduct pipeline for each product.

2.4.3 Sync Current

The Sync Current button runs the saveInsiteProduct pipeline for the currently selected item. If the item does not correspond to an Insite product, an error message is displayed.

Chapter 3   Insite API & Sitecore

While product synchronization exposes Insite products as Sitecore items, it is not a replacement for the Insite API. This chapter describes some enhancements that make it easier to work with the Insite API from Sitecore code.

3.1 Context Product

Product information is displayed on web pages using Sitecore renderings. The idea behind the context product is very similar to Sitecore's context item: it represents the current product.

The following code demonstrates how to determine the current Insite product:

```
var product = Sitecore.Marketing.Commerce.Insite.CurrentProduct;
```

Insite Connect uses the pipeline getInsiteProductForContextItem to determine the current product.

Chapter 4   SES API Extensions

SES includes an API that can be extended in order to integrate information from external systems. Insite Connect includes a number of extensions to this API. The purpose is to use Insite to manage certain information rather than Sitecore.

An example of this is product price information. Insite provides a much more powerful system for managing prices. The SES API makes it possible to replace the component responsible for determining a product's price without requiring major changes to SES.

4.1 Product Stock Manager

The Product Stock Manager is the SES component that is responsible for determining the inventory level of a specific product. The Product Stock Manager that ships with SES uses a field on the Sitecore product item to store the product stock level. Insite Connect provides a Product Stock Manager that uses Insite to determine the inventory level.

4.1.1 Configuration

In order to use the custom Product Stock Manager the Unity.config file must be updated.

```
<alias alias="ProductStockManager"
       type="Sitecore.Marketing.Commerce.Insite.Inventory.InsiteProductStockManager,
             Sitecore.Marketing.Commerce.Insite"/>
```

4.2 Product Price Manager

The Product Price Manager is the SES component that is responsible for determining the price of a specific product. The Product Price Manager that ships with SES uses fields on the Sitecore product item to store the product price. Insite Connect provides a Product Price Manager that uses Insite to determine the price.

4.2.1   Configuration

In order to use the custom Product Price Manager the Unity.config file must be updated.

```
<alias alias="ProductPriceManager"
       type="Sitecore.Marketing.Commerce.Insite.Pricing.PipelineProductPriceManager,
             Sitecore.Marketing.Commerce.Insite"/>
```

4.3 Shopping Cart Manager

The Shopping Cart Manager is the SES component that is called when items are added to and removed from the shopping cart. The Shopping Cart Manager that ships with SES stores the items added to a cart in-memory. Insite Connect provides a Shopping Cart Manager extends the default manager so items are also added to/removed from the Insite shopping cart.

4.3.1 Configuration

In order to use the custom Product Price Manager the Unity.config file must be updated.

```
<alias alias="ShoppingCartManager"
       type="Sitecore.Marketing.Commerce.Carts.PipelineShoppingCartManager,
             Sitecore.Marketing.Commerce.Insite"/>
```

Chapter 5 **Web Forms For Marketers**

Web Forms For Marketers makes it possible for business users to create interactive forms without involving a developer. This chapter describes how Insite Connect integrates with Web Forms For Marketers.

5.1 Commerce Login Save Action

When a visitor logs into a site, Insite needs to know the user has logged in. The Commerce Login save action incorporates the extra logic required to support this.

The save action triggers the pipeline commerceLogin.

Chapter 6 DMS Integration

Sitecore DMS provides a large number of tracking and visitor analysis tools. Insite Connect hooks into the existing tools in order to provide some commerce-specific tracking capabilities.
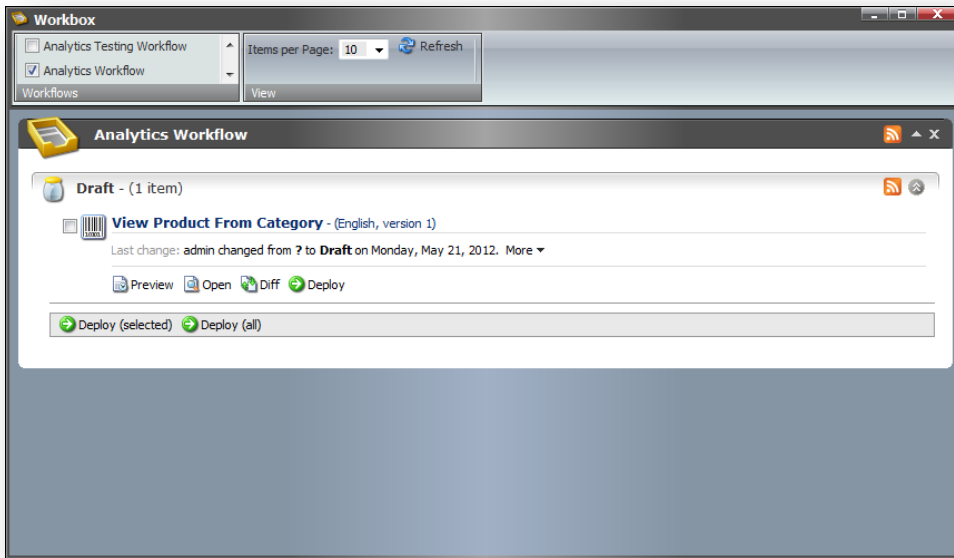
6.1 Page Events

Insite Connect includes custom page events that track commerce-specific behavior.

6.1.1 Configuration

The installation package creates new Page Event definition items for DMS. These items must be deployed (using Sitecore's workflow features) and published:

- View Product From Category

6.1.2 View Product From Category

The View Product From Category page event can record the fact that a visitor has viewed a product from a category. This is an important way to be able to identify trends and patterns.

This page event does not get triggered automatically. It must be configured using a Product Item View Rule.

6.2 Product Item View Rules

A Product Item View Rule is triggered every time a product item is viewed.

6.2.1   Extending Templates To Support Product Item View Rules

The rules are triggered by a processor in the renderLayout pipeline. This processor looks for a field named "Item View Rules" on the item. If the item has this field then the rules will be run at render-time.

In order to extend an existing template to support Product Item View Rules you must add a base template to any templates used for products. The base template is located at /sitecore/templates/Insite/Item View Rules.