Tealium Sitecore Tag Management Module Specification

Module Version 1.0.2.0

Contents

1.0 Overview

This document describes how to install, configure and integrate the **"Tealium.Sitecore.TagManagement"** Sitecore module into your environment.

1.1 Module Description

The module represents an object-oriented framework that allows you to enable the Tealium Tag Management scripts in your solution. The module is fully configurable and customizable.

1.2 Compatibility

The **"Tealium.Sitecore.TagManagement"** Sitecore module is compatible with following:

- Sitecore CMS version 6.0 and upper
- .NET Framework 4.0, 4.5

2.0 Installation

The **"Tealium Sitecore Tag Management"** module is a well-formed Sitecore module and should be installed via the default Sitecore installation wizard.

2.1 Module Contents

The **"Tealium Sitecore Tag Management"** module installer adds the following files and items into your website:

2.1.1 Items

- /sitecore/templates/Tealium
    - /sitecore/templates/Tealium/Settings
        - /sitecore/templates/Tealium/Settings/Tealium Settings
        - /sitecore/templates/Tealium/Settings/Utag Data Mapping Settings
        - /sitecore/templates/Tealium/Settings/Utag Data Template Mapping
        - /sitecore/templates/Tealium/Settings/Entities
        - /sitecore/templates/Tealium/Settings/Entities/Option
        - /sitecore/templates/Tealium/Settings/Entities/Option Folder
    - /sitecore/templates/Tealium/Sample Data
        - /sitecore/templates/Tealium/Sample Data/Product Page
- /sitecore/system/Modules/Tealium Tag Management
    - /sitecore/system/Modules/Tealium Tag Management/Settings
        - /sitecore/system/Modules/Tealium Tag Management/Settings/Tealium Settings
        - /sitecore/system/Modules/Tealium Tag Management/Settings/Utag Data Mapping Settings
    - /sitecore/system/Modules/Tealium Tag Management/Entities
        - /sitecore/system/Modules/Tealium Tag Management/Entities/Dropdown Options

2.1.2 Files

- /bin/Tealium.Sitecore.TagManagement.dll
- /bin/Tealium.Sitecore.TagManagement.pdb
- /bin/net40/Release/Tealium.Sitecore.TagManagement.dll
- /bin/net40/Release/Tealium.Sitecore.TagManagement.pdb
- /App_Config/Include/Tealium.Sitecore.TagManagement.config
- /Sitecore/shell/Themes/Standard/Tealium.zip

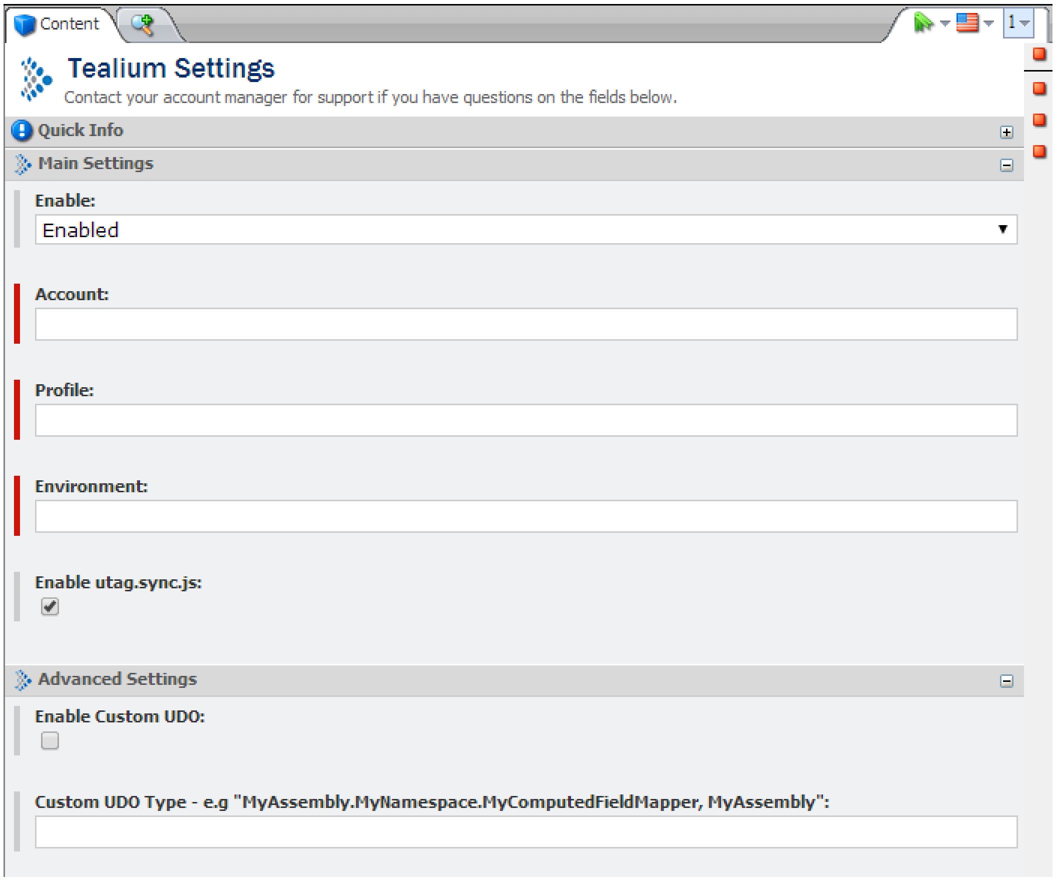
2.2 Installation Process

1. In order to install the module, please download the module zip package.
2. Please open your Sitecore instance.
3. Please go to **"Sitecore -> Development Tools -> Installation Wizard"**.
4. Please upload the package into your Sitecore and click to install.

3.0 Configuration and Tuning

3.1 Main Settings

Once the module is installed, you can configure the main settings by changing fields on the settings screen **"/sitecore/system/Modules/Tealium Tag Management/Settings/Tealium Settings"**:



Tealium Settings

Contact your account manager for support if you have questions on the fields below.

Quick Info

Main Settings

Enable:
Enabled ▾

Account:

Profile:

Environment:

Enable utag.sync.js:
☑

Advanced Settings

Enable Custom UDO:
□

Custom UDO Type - e.g "MyAssembly.MyNamespace.MyComputedFieldMapper, MyAssembly":

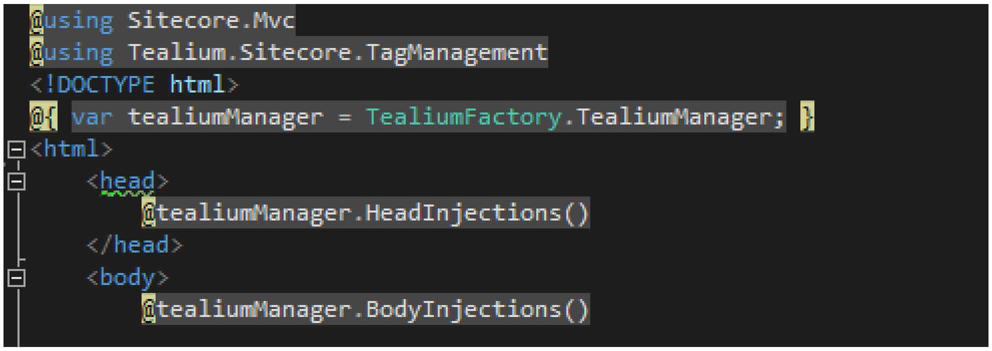
- **Main Settings:**
  - **Enable**: This field allows you to enable or completely disable the module on the website without making any changes to the source code. The **“Enabled”** setting adds the core scripts and data layer scripts to the website. The **“Disabled”** setting removes the core scripts and data layer scripts from the website.
  - **Account**: This field allows you to enter your Tealium account name. This field is mandatory.
  - **Profile**: This field allows you to enter your Tealium profile name. This field is mandatory.
  - **Environment**: This field allows you to enter your Tealium environment name. This field is mandatory.
  - **Enable utag.sync.js**: This field allows you to enable or disable the **“utag.sync.js”** script on the website. If this checkbox is checked, the module adds the script to the head section of the website pages; otherwise, the module removes the script from the head section of the website pages.

- **Advanced Settings:**
  - **Enable Custom UDO**: This field allows you to extend the data layer with additional name / value pairs without making any changes to the module source code.
  - **Custom UDO Type**: This field allows you to specify an assembly / type name in order to extend the default data layer functionality. This works only when the **“Enable Custom UDO”** checkbox is checked. For more information about extending the default data layer functionality please read about computed fields later in this specification.

Important: After you specify all the above settings, please do not forget to save and publish your changes to the target database.

3.2 Layout Injections

Once you complete configuring your Tealium settings, please add some pieces of code into your layout file. Please go to your layout file and create an instance of the **“TealiumManager”** class. Please note that in order to create an instance of the **“TealiumManager”** class you should use the **“TealiumFactory”** class. After you instantiate it, please call the following 2 methods: the first one inside the **“<head>”** section, the second one right after opening the **“<body>”** section:



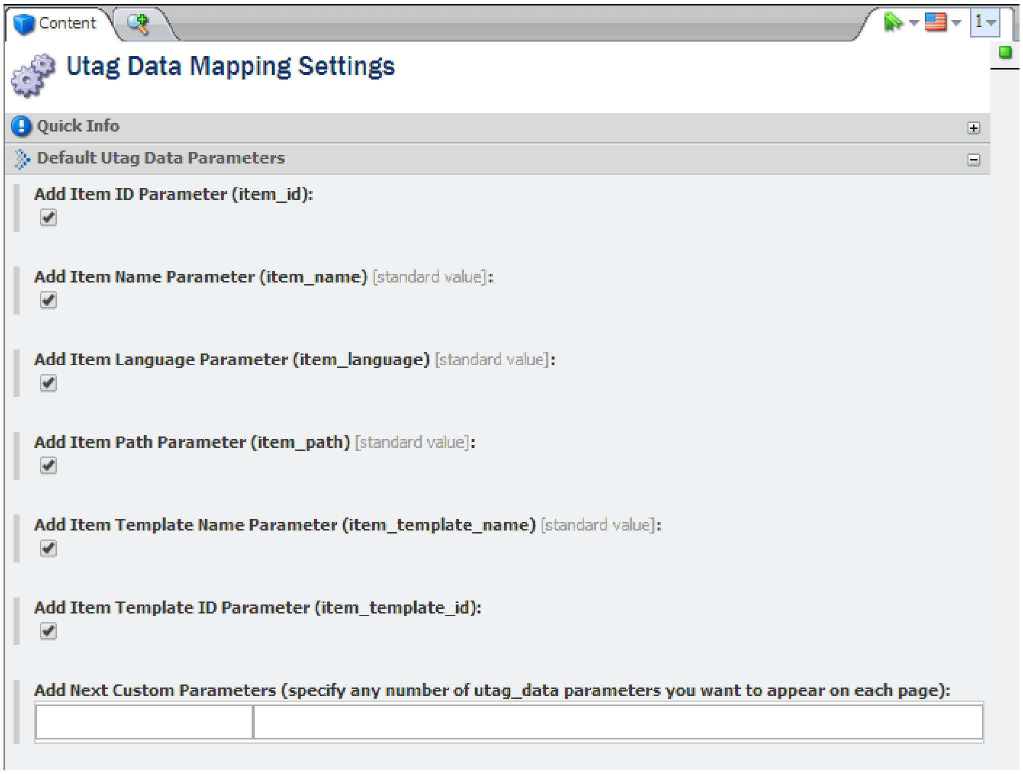
```
@using Sitecore.Mvc
@using Tealium.Sitecore.TagManagement
<!DOCTYPE html>
@{ var tealiumManager = TealiumFactory.TealiumManager; }
<html>
    <head>
        @tealiumManager.HeadInjections()
    </head>
    <body>
        @tealiumManager.BodyInjections()
```

3.3 Utag Data Settings

This item allows you to configure what data goes to the **"utag_data"** JavaScript object inside the generated script.

3.3.1 Common Utag Data

In order to configure common parameters that should be present on all page types, please open the **"/sitecore/system/Modules/Tealium Tag Management/Settings/Utag Data Mapping Settings"** screen:



Utag Data Mapping Settings

Quick Info

Default Utag Data Parameters

Add Item ID Parameter (item_id):
☑

Add Item Name Parameter (item_name) [standard value]:
☑

Add Item Language Parameter (item_language) [standard value]:
☑

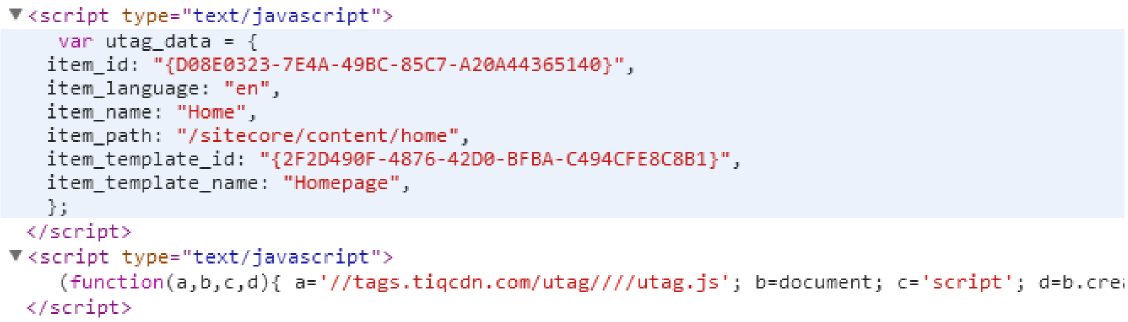
Add Item Path Parameter (item_path) [standard value]:
☑

Add Item Template Name Parameter (item_template_name) [standard value]:
☑

Add Item Template ID Parameter (item_template_id):
☑

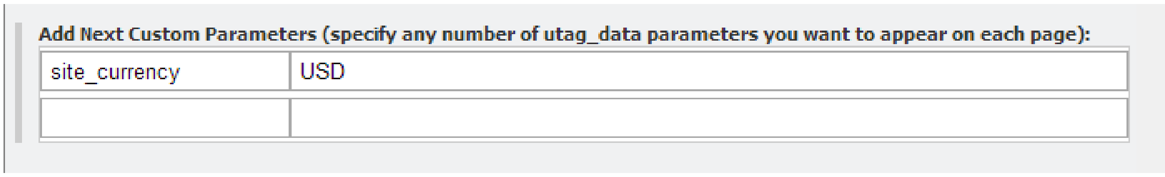
Add Next Custom Parameters (specify any number of utag_data parameters you want to appear on each page):

Please check a checkbox if you would like the module to render a corresponding parameter within the **"utag_data"** JavaScript object on all page types. Please see a sample output below:



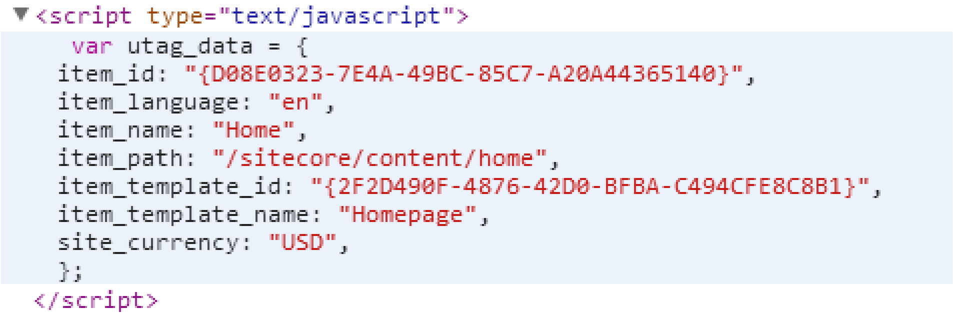
```
▼<script type="text/javascript">
    var utag_data = {
   item_id: "{D08E0323-7E4A-49BC-85C7-A20A44365140}",
   item_language: "en",
   item_name: "Home",
   item_path: "/sitecore/content/home",
   item_template_id: "{2F2D490F-4876-42D0-BFBA-C494CFE8C8B1}",
   item_template_name: "Homepage",
   };
 </script>
▼<script type="text/javascript">
    (function(a,b,c,d){ a='//tags.tiqcdn.com/utag///utag.js'; b=document; c='script'; d=b.cre
 </script>
```

In order to make the module render a custom name / value pair within the **"utag_data"** JavaScript object across all page types, please use the **"Add Next Custom Parameters"** functionality. Please see an example below:



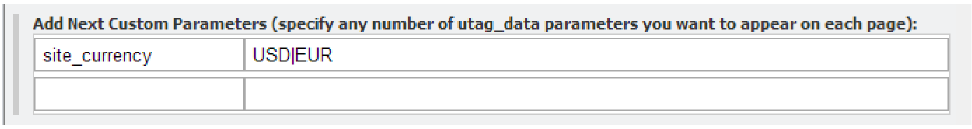
Add Next Custom Parameters (specify any number of utag_data parameters you want to appear on each page):

site_currency	USD



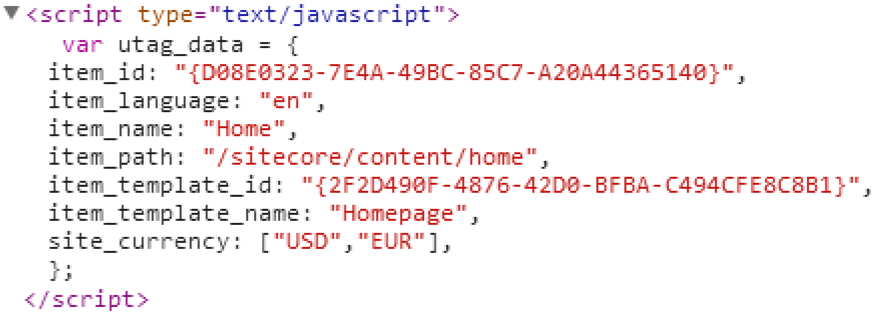
```
▼<script type="text/javascript">
   var utag_data = {
  item_id: "{D08E0323-7E4A-49BC-85C7-A20A44365140}",
  item_language: "en",
  item_name: "Home",
  item_path: "/sitecore/content/home",
  item_template_id: "{2F2D490F-4876-42D0-BFBA-C494CFE8C8B1}",
  item_template_name: "Homepage",
  site_currency: "USD",
  };
 </script>
```

In order to specify multiple values for a single parameter, please use the default Sitecore delimiter which is "|" or a comma. In this case, all specified values will be rendered as a JSON array. Please see an example below:



Add Next Custom Parameters (specify any number of utag_data parameters you want to appear on each page):

site_currency	USD|EUR

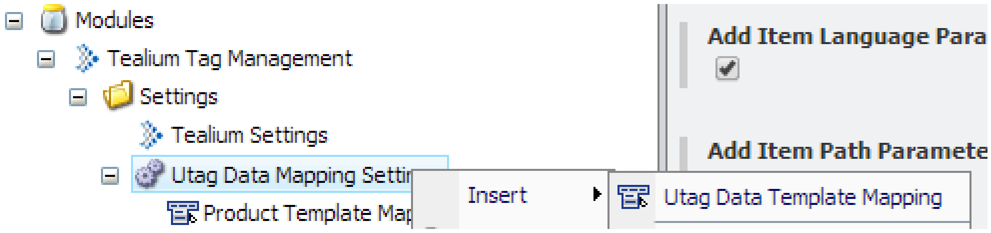


```
▼<script type="text/javascript">
   var utag_data = {
  item_id: "{D08E0323-7E4A-49BC-85C7-A20A44365140}",
  item_language: "en",
  item_name: "Home",
  item_path: "/sitecore/content/home",
  item_template_id: "{2F2D490F-4876-42D0-BFBA-C494CFE8C8B1}",
  item_template_name: "Homepage",
  site_currency: ["USD","EUR"],
  };
 </script>
```

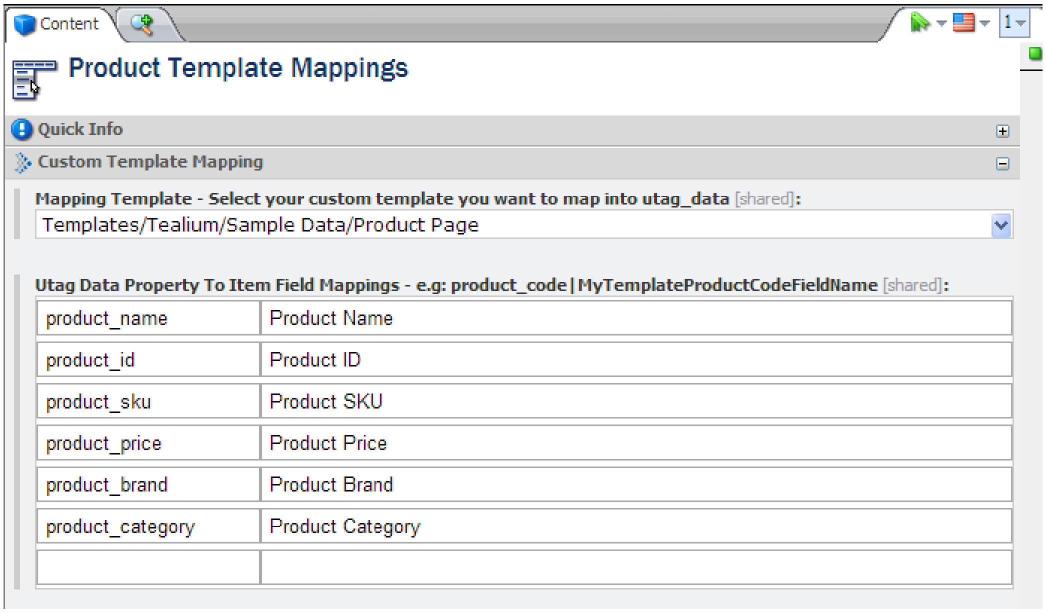
Important: After you configure all the above parameters, please do not forget to save and publish your changes to the target database.

3.3.2 Utag Data Mappings

The module allows you to create any number of data template mappings inside **"/sitecore/system/Modules/Tealium Tag Management/Settings/Utag Data Mapping Settings"**:



A data template mapping is an association between a page template and data properties which should be rendered within the **“utag_data”** JavaScript object on all pages created based on this page template or based on any page templates inherited this page template. Please see an example below:



Product Template Mappings

Quick Info

Custom Template Mapping

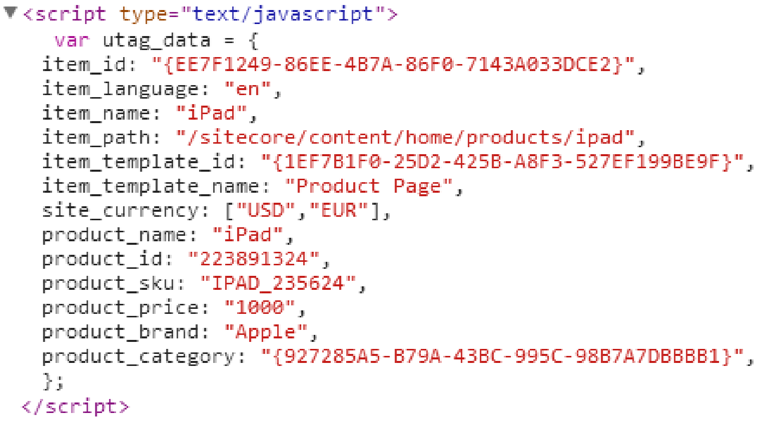
Mapping Template - Select your custom template you want to map into utag_data [shared]:

Templates/Tealium/Sample Data/Product Page

Utag Data Property To Item Field Mappings - e.g: product_code | MyTemplateProductCodeFieldName [shared]:

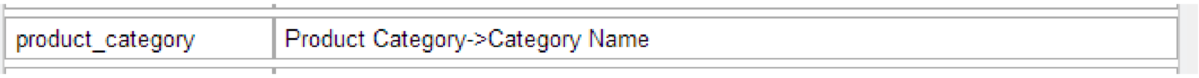
product_name	Product Name
product_id	Product ID
product_sku	Product SKU
product_price	Product Price
product_brand	Product Brand
product_category	Product Category

The **“Mapping Template”** field allows you to select a page template. The “**Utag Data Property To Item Field Mappings**” field allows you to map a page template field to a desired **“utag_data”** parameter. Please see a sample output below:

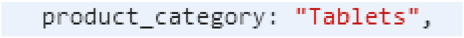


```
▼<script type="text/javascript">
   var utag_data = {
  item_id: "{EE7F1249-86EE-4B7A-86F0-7143A033DCE2}",
  item_language: "en",
  item_name: "iPad",
  item_path: "/sitecore/content/home/products/ipad",
  item_template_id: "{1EF7B1F0-25D2-425B-A8F3-527EF199BE9F}",
  item_template_name: "Product Page",
  site_currency: ["USD","EUR"],
  product_name: "iPad",
  product_id: "223891324",
  product_sku: "IPAD_235624",
  product_price: "1000",
  product_brand: "Apple",
  product_category: "{927285A5-B79A-43BC-995C-98B7A7DBBBB1}",
  };
 </script>
```

If you have a linked field in your data template mapping (like **“Product Category”** from the sample above), you might use the **“->”** delimiter in order to display any other information from a linked entity (for example, a category name instead of its ID). Please see an example below:

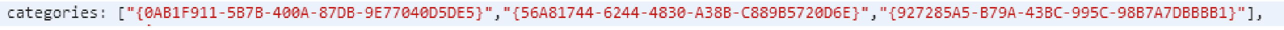


product_category	Product Category->Category Name

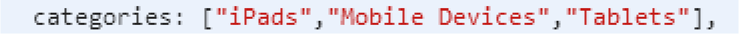


```
  product_category: "Tablets",
```

The same syntax is also applicable to any linked field types with multiple values (such as multilist or treelist). Please see an example below:



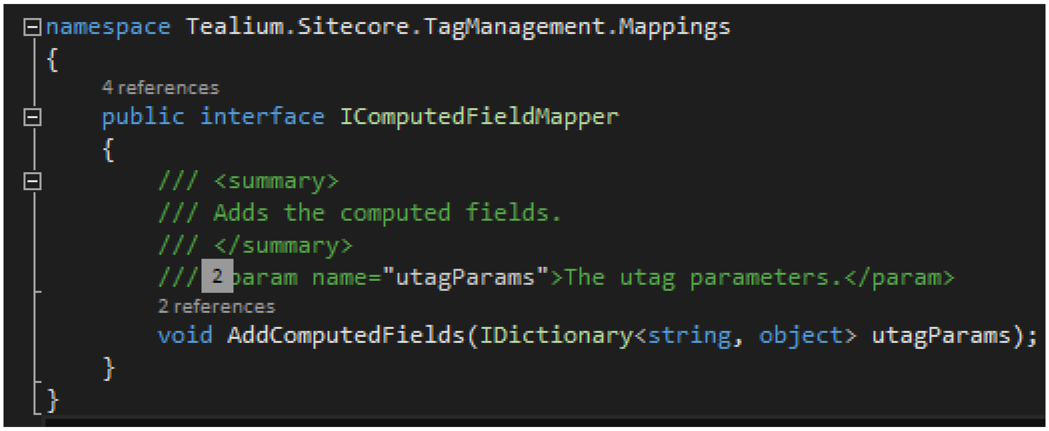
```
categories: ["{0AB1F911-5B7B-400A-87DB-9E77040D5DE5}","{56A81744-6244-4830-A38B-C889B5720D6E}","{927285A5-B79A-43BC-995C-98B7A7DBBBB1}"],
```



```
categories: ["iPads","Mobile Devices","Tablets"],
```

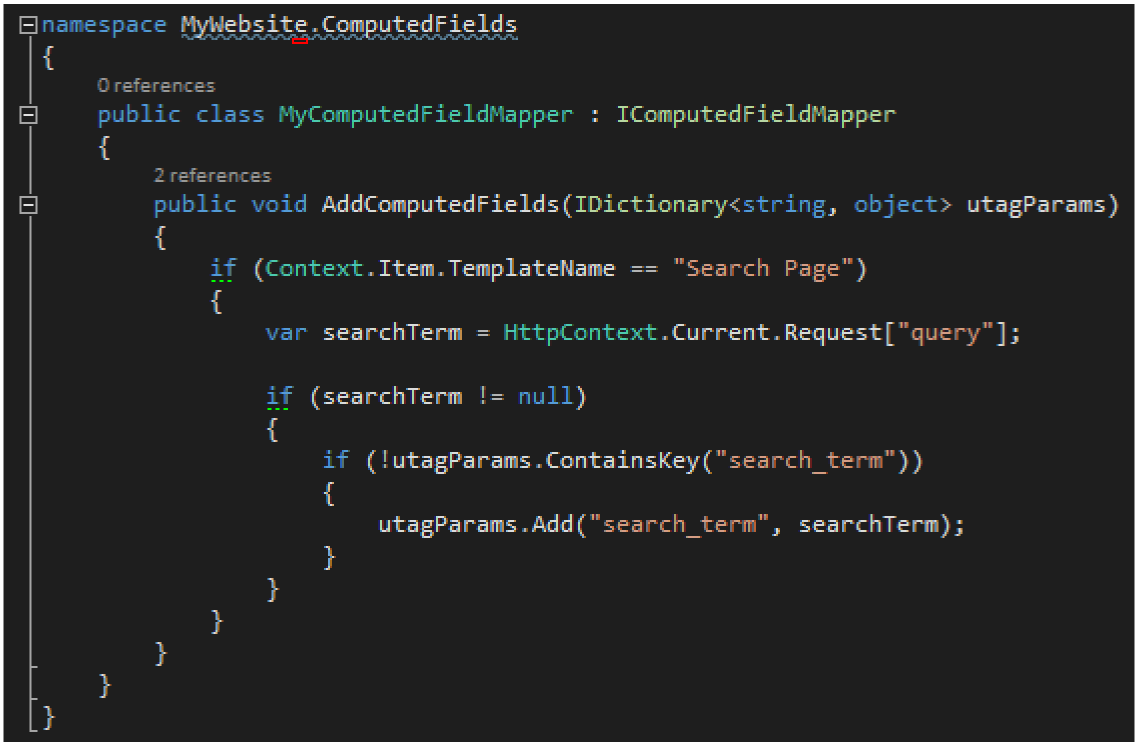
3.3.3 Custom UDO and Computed Field Mappings

The module allows you to implement a custom field mapping for any page template. In order to implement a custom field mapping, you should inherit the **"IComputedFieldMappings"** interface. Please its code below:



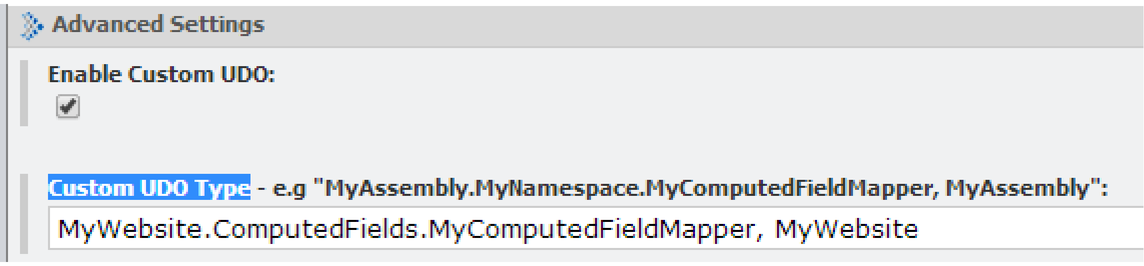
```
namespace Tealium.Sitecore.TagManagement.Mappings
{
    4 references
    public interface IComputedFieldMapper
    {
        /// <summary>
        /// Adds the computed fields.
        /// </summary>
        /// <param name="utagParams">The utag parameters.</param>
        2 references
        void AddComputedFields(IDictionary<string, object> utagParams);
    }
}
```

Also, please see below a possible implementation of adding a search term into a search page template:



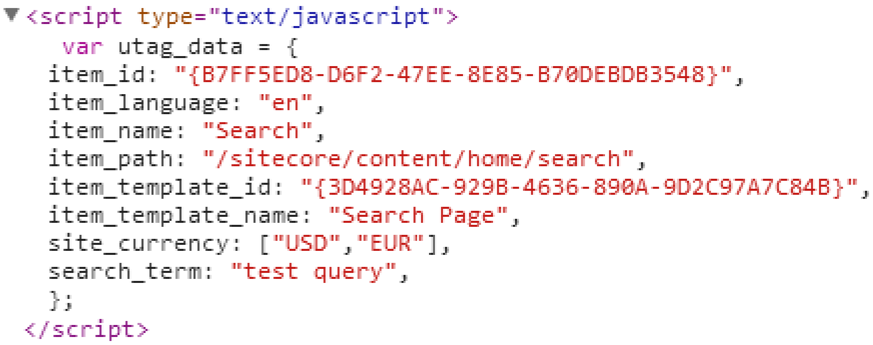
```
namespace MyWebsite.ComputedFields
{
    0 references
    public class MyComputedFieldMapper : IComputedFieldMapper
    {
        2 references
        public void AddComputedFields(IDictionary<string, object> utagParams)
        {
            if (Context.Item.TemplateName == "Search Page")
            {
                var searchTerm = HttpContext.Current.Request["query"];

                if (searchTerm != null)
                {
                    if (!utagParams.ContainsKey("search_term"))
                    {
                        utagParams.Add("search_term", searchTerm);
                    }
                }
            }
        }
    }
}
```

After you implement your custom field mapping, please do not forget to check the "**Enable Custom UDO**" checkbox on the module settings screen as well as specify a fully-qualified type name in the **"Custom UDO Type"** input box (please see the main settings section for more details). Please see an example below:



Enable Custom UDO:
☑

Custom UDO Type - e.g "MyAssembly.MyNamespace.MyComputedFieldMapper, MyAssembly":
MyWebsite.ComputedFields.MyComputedFieldMapper, MyWebsite



```
▼<script type="text/javascript">
    var utag_data = {
   item_id: "{B7FF5ED8-D6F2-47EE-8E85-B70DEBDB3548}",
   item_language: "en",
   item_name: "Search",
   item_path: "/sitecore/content/home/search",
   item_template_id: "{3D4928AC-929B-4636-890A-9D2C97A7C84B}",
   item_template_name: "Search Page",
   site_currency: ["USD","EUR"],
   search_term: "test query",
   };
 </script>
```

3.3.4 Configuration File

The module has its configuration file which can be found by the following path – **"/App_Config/Include/Tealium.Sitecore.TagManagement.config"**. This configuration file has the following configuration settings:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
 <sitecore>
  <events>
   <event name="publish:end">
    <handler type="Tealium.Sitecore.TagManagement.Events.PublishEnd, Tealium.Sitecore.TagManagement"
method="Refresh" />
   </event>
   <event name="publish:end:remote">
    <handler type="Tealium.Sitecore.TagManagement.Events.PublishEnd, Tealium.Sitecore.TagManagement"
method="Refresh" />
   </event>
   <event name="publish:complete">
    <handler type="Tealium.Sitecore.TagManagement.Events.PublishEnd, Tealium.Sitecore.TagManagement"
method="Refresh" />
   </event>
   <event name="publish:complete:remote">
    <handler type="Tealium.Sitecore.TagManagement.Events.PublishEnd, Tealium.Sitecore.TagManagement"
method="Refresh" />
   </event>
  </events>
```

```
    <settings>
      <setting name="Tealium.DataProvider" value="Tealium.Sitecore.TagManagement.Data.UtagDataProvider,
Tealium.Sitecore.TagManagement"></setting>
      <setting name="Tealium.MappingsProvider"
value="Tealium.Sitecore.TagManagement.Mappings.TealiumMappingsProvider,
Tealium.Sitecore.TagManagement"></setting>
      <setting name="Tealium.TealiumSettings.ItemID" value="{A5F3FD05-62D4-468A-BB54-
6C9760F701A5}"></setting>
      <setting name="Tealium.UtagDataMappingSettings.ItemID" value="{D77D079B-6A79-490D-B4F5-
408D773FC3E8}"></setting>
      <setting name="Tealium.Utag.Js.UriFormat" value="//tags.tiqcdn.com/utag/{0}/{1}/{2}/utag.js"></setting>
      <setting name="Tealium.Utag.Sync.Js.UriFormat"
value="//tags.tiqcdn.com/utag/{0}/{1}/{2}/utag.sync.js"></setting>
    </settings>
  </sitecore>
</configuration>
```

- **Publish End/Complete** – the SettingsProvider and the MappingsProvider cache the settings to improve the module performance and after each publishing it's required to clear their caches for new data values are successfully applied.
- **Tealium.DataProvider** – the qualified type name for the DataProvider. This allows developers to write custom implementation of the data provider. However, all functionality that needs to be customized should be available by implementing custom field mappings.
- **Tealium.MappingsProvider** – the qualified type name for the MappingsProvider. This allows developers to write custom implementation of mappings provider. However, all functionality that needs to be customized should be available by implementing custom field mappings.
- **Tealium.TealiumSettings.ItemID** – the settings item id that the module uses for reading main settings. You could change it in the case of creating the new settings item.
- **Tealium.UtagDataMappingSettings.ItemID** – the mappings settings item id that the module uses for reading mappings settings. You could change it in the case of creating the new mappings settings item.

4.0 Version 1.0.2.0 Overview

This section describes a list of new features added in version 1.0.2.0 of the **“Tealium.Sitecore.TagManagement”** Sitecore module.

4.1 Description

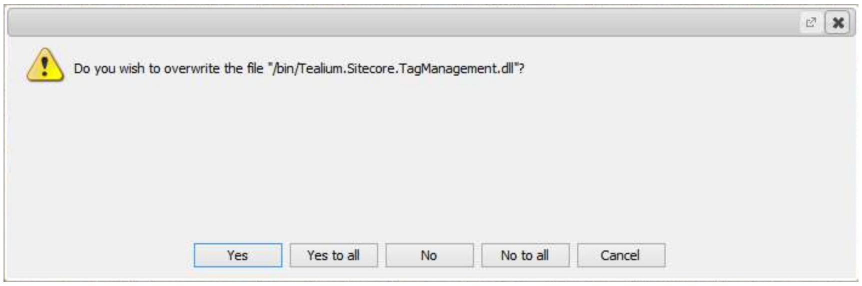
The main and only feature added in version 1.0.2.0 is a possibility to configure different settings and mappings for different Sitecore websites in a single solution.

4.2 Installation

The **“Tealium Sitecore Tag Management”** module is a well-formed Sitecore module and should be installed via the default Sitecore installation wizard. However, if you would like to update from version 1.0.0.3, please read the instruction below.

4.2.1 Update From 1.0.0.3 To 1.0.2.0

In order to update the module from 1.0.0.3, please download the **“Tealium Tag Management-1.0.2.0-update.zip”** package and follow common instructions described in the installation process section at the beginning of the document. The main difference between the update package and the package for the clean installation is that the update package does not override any of your previously configured settings. If during the installation process the system asks you to merge conflicted files, please choose **“Override”** to all the conflicts found. Please see an example below:



Do you wish to overwrite the file "/bin/Tealium.Sitecore.TagManagement.dll"?

4.3 Module Contents

Version 1.0.2.0 contains all items and files included into the first version of the module (please read the module contents section at the beginning of the document) as well as some additional ones. The both sections below list only additional items and files added in version 1.0.2.0.

4.3.1 Items

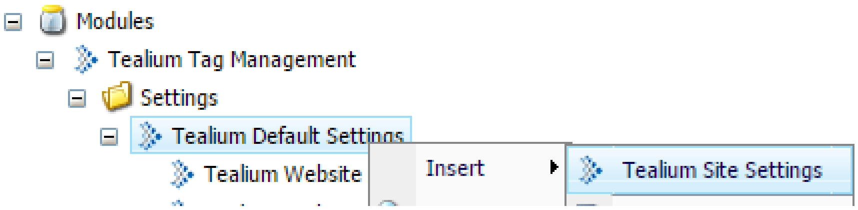
- master: sitecore/templates/Tealium/Settings/Tealium Site Settings
- master: /sitecore/templates/Tealium/Settings/Utag Data Site Mapping Settings
- core: /sitecore/system/Field types/Tealium
    - core: /sitecore/system/Field types/Tealium/Site Selector Droplist

4.3.2 Files

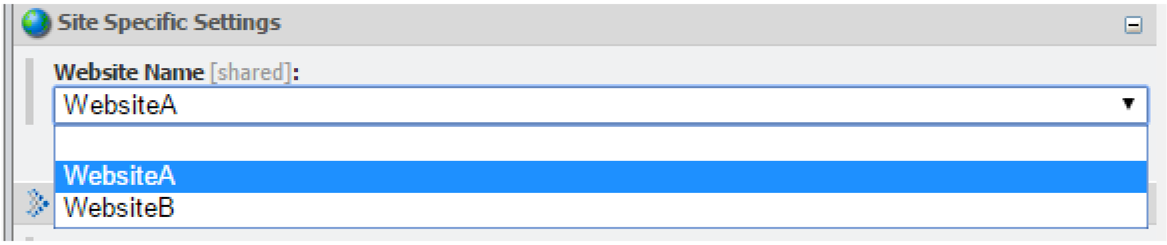
- \App_Config\Include\Tealium.Sitecore.TagManagement.CustomFields.config

4.4 Configuration and Tuning

In case you update your 1.0.0.3 version to 1.0.2.0, your module settings element **"Tealium Settings"** will be renamed to **"Tealium Default Settings"** and will be used as default settings for all non-customized websites (websites that do not have their own Tealium settings). Please see an example below:



The **"Tealium Default Settings"** element allows you to create specific settings for a particular website. Please see an example below:



Website Name [shared]:
WebsiteA

WebsiteA
WebsiteB

In order to create custom site settings for a specific website, please follow the steps below:

1. Please make sure your website(s) is configured in the **"<sites>"** node in the Sitecore configuration.
2. Please create a site specific settings item under the default settings item and configure all necessary fields.
3. The **"Website Name"** field will automatically list all available websites configured in the Sitecore configuration file in the "<sites>" node.

```
<sites>
        <site name="WebsiteA" hostName=… />
        <site name="WebsiteB" hostName=… />
</sites>
```

4. Please select a website you want, save and publish your changes.
5. If you have other websites in your solution and they do not have custom settings, the module will use the default settings configured in **"Tealium Default Settings"**.
6. If you are configuring Tealium settings for just another language of a particular website and utilizing the same content tree, please update Tealium settings for that specific language instead of creating a site specific settings item.

4.4.1 Website Name Field Configuration

The Sitecore solution usually has a huge number of predefined websites configured in the **<sites>** node in the Sitecore configuration file. Almost all of those predefined websites are system websites and should not be displayed in the **"Website Name"** drop-down box. In order to manage a list of websites which should not be visible, please use the "**Tealium.CustomFields.SitesToIgnore**" configuration setting in the

“**\App_Config\Include\Tealium.Sitecore.TagManagement.CustomFields.config**” configuration file. The following websites are hidden by default:

```
<setting name="Tealium.CustomFields.SitesToIgnore"
value="admin|login|modules_shell|modules_website|publisher|scheduler|service|shell|system|website"></setting>
```

4.5 Utag Data Mappings

In case you update your 1.0.0.3 version to 1.0.2.0, your module settings element **“Utag Data Mapping Settings”** will be renamed to **“Utag Data Mapping Default Settings”** and will be used as default settings for all non-customized websites (websites that do not have their own data mapping settings). If you would like to create site specific data mapping settings, please do it under the default data mapping element using the **“Website Name”** field.