

## **Sense/Net 6 Evaluation Guide**

*How to build a products listing site from the scratch?*

## Contents

1	Basic principles.....	4
1.1	Everything is a content .....	4
1.2	Pages .....	5
1.3	Smart application model .....	5
1.4	Queries.....	5
1.5	Skin system .....	6
2	Let's build a site! .....	7
2.1	Define a content type .....	7
2.2	Define a content view .....	10
2.3	Define the basic structure .....	13
2.4	Put portlets on the page.....	14
2.4.1	Simple static content presentation .....	15
2.4.2	Query.....	17
2.5	A word on navigation .....	22
2.6	Some basic design stuff .....	23

## *Foreword*

Welcome to Sense/Net and thank you for your interest in downloading our open-source, Enterprise Content Management System.

This evaluation guide will take you through the process of building a simple site with Sense/Net from the scratch. You can do much more with it, though – complex Intranets, enterprise workflows, document management systems, collaborative portals. If you're looking for deeper knowledge in these areas, please contact us for training possibilities.

This first step for building a simple site is necessary for understanding the basic principles and logic of Sense/Net.

If you have any problems installing Sense/Net, take a look at this Wiki article: [http://wiki.sensenet.com/index.php?title=How\\_to\\_install\\_Sense/Net](http://wiki.sensenet.com/index.php?title=How_to_install_Sense/Net).

Additional Sense/Net information can be found at <http://wiki.sensenet.com>.

## 1 Basic principles

Here we'll cover some of the basic principles for your evaluation. Not even half of the entire system is described by this, but it'll do for a basic evaluation. Contact us, if you need further assistance.

### 1.1 Everything is a content

First and foremost what you need to understand that everything is a content in Sense/Net. If you open Explore view and take a look around in the default hierarchy, you'll find plenty of objects there. Some are containers, some are list items, some are articles, some are CSS or ASCX files. From the system's perspective, they're all content.

This means, that they are described by a content type definition (in short: CTD) object (which is a content itself). They have fields (in which data is stored) and they have properties (like permissions, lock, state, validity, etc.). If you create a new content from a given content type, you need to fill the defined fields. The CTD is basically an XML.

Some existing content types (there's a lot, so these are just some examples):

- article
- user
- list item
- forum topic
- group
- workflow
- tag
- folder
- file

Content types have inheritance, so if you create a new content type under "Folder", it will inherit its settings.

You can see the content types here in your system:

<http://<<yoursiteURL>>/Explore.html#/Root/System/Schema/ContentTypes>

More about content types:

[http://wiki.sensenet.com/index.php?title=Content\\_Type](http://wiki.sensenet.com/index.php?title=Content_Type)

## 1.2 Pages

Pages are special type of content. They show one or more portlets to users. A portlet is a smaller part of the page, presenting content, application, a list of contents, etc. A page can be edited from the site by adding, removing or re-arranging portlets.

Every page can have different design and layout settings (selected at the page's property settings).

More about pages:

<http://wiki.sensenet.com/index.php?title=Page>

## 1.3 Smart application model

Every type of content is capable of rendering itself. This means that they are addressable and if you enter their URL, they will be displayed on the screen by their Browse application. If you edit them, their Edit application will display them. There are many applications like Delete, CheckOut, History, RSS, etc – we're not covering this here.

The father of all content types, the Generic Content has its own default renderers. If you do not specify your own for your own custom content type, it will use the Generic Content's applications when addressed.

You may also reference an application for a given content by URL – for example, editing a piece of content can be achieved by going to the URL

<http://<<yourSite>>/<<yourpath>>/<<yourcontent>>?Action=Edit>. Of course only users with valid edit permission may access this URL.

More about the smart application model:

[http://wiki.sensenet.com/index.php?title=Smart\\_Application\\_Model](http://wiki.sensenet.com/index.php?title=Smart_Application_Model)

## 1.4 Queries

80% of the portal building work you'll work with queries. A query is a near-humane language search term that gives you back a content or a set of contents which you can display on the site. There are several built-in portlets that can do this trick for you (Content Query Portlet), but in a short time you'll probably need your own views to represent the results (think of a news collection, latest blog entries, product catalogue, etc.).

More about queries:

[http://wiki.sensenet.com/index.php?title=Query\\_syntax](http://wiki.sensenet.com/index.php?title=Query_syntax)

## 1.5 Skin system

Skin system is the design and layout module within Sense/Net.

You can define the layout of the sites by defining ASP.NET Master Page templates. This defines zones, into which portlets can be inserted.

Design is set via CSS files.

More about the skin system:

[http://wiki.sensenet.com/index.php?title=The\\_Skin\\_System](http://wiki.sensenet.com/index.php?title=The_Skin_System)

## 2 Let's build a site!

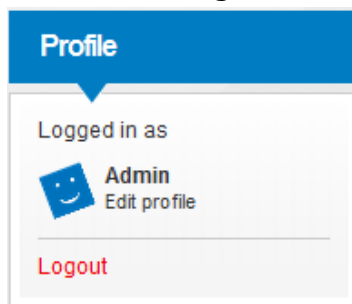
We'll assume that you could install Sense/Net and you want to have a first-time experience by building a couple of pages. Our sample site will be a products catalogue site. We'll create a new content type, upload some products, create a product listings page and a product page. We'll also insert some static content to greet our users, some navigation and we'll modify the Sense/Net skin to change the design a bit.

### 2.1 Define a content type

Do the following:

1. Log in to the system as an administrator. If you haven't changed anything since installation, the login portlet is on the left of the opening page. The user name/password is: admin/admin.

*You should see this after a successful login:*



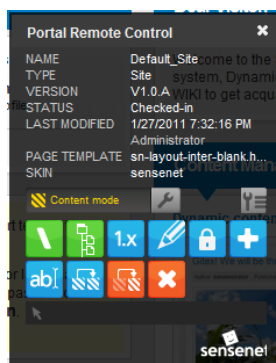
2. Please take note of the small icon appearing on the top right of your screen. This is the button to open the portal remote control.

*If should look like this:*

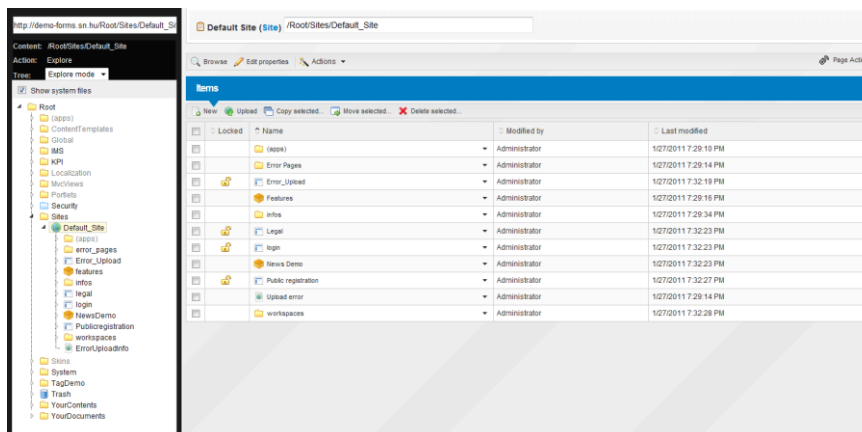


3. Click on that icon! The portal remote control appears.

*It looks like this:*



- Open the Explore View by clicking on the folder-tree like icon (the second green one). You switch to Explore View. If you need to see the site while manipulating the Content Repository (the huge thing with the tree on the left which just appeared), we suggest to open a new tab with the site. That will be easier.



- Locate the /Root/System/Schema/ContentTypes/GenericContent/Folder container on the left and open it. Our Product content type will be the child of the Folder content type.
- Create a new content type by clicking New.
- Enter (Copy-Paste) this:

```
<ContentType name="Product" parentType="Folder"
handler="SenseNet.ContentRepository.GenericContent"
xmlns="http://schemas.sensenet.com/SenseNet/ContentRepository/Content
TypeDefinition">
  <DisplayName>Product</DisplayName>
  <Description>A type for products</Description>
  <Icon>FormItem</Icon>
  <Fields>
    <Field name="DisplayName" type="ShortText">
      <DisplayName>Display Name</DisplayName>
      <Configuration>
        <VisibleBrowse>Show</VisibleBrowse>
        <VisibleEdit>Show</VisibleEdit>
        <VisibleNew>Show</VisibleNew>
      </Configuration>
    </Field>
    <Field name="Name" type="ShortText">
      <Configuration>
        <VisibleBrowse>Show</VisibleBrowse>
        <VisibleEdit>Show</VisibleEdit>
        <VisibleNew>Show</VisibleNew>
      </Configuration>
    </Field>
  </Fields>
</ContentType>
```

```

</Field>
<Field name="Vendor" type="ShortText">
  <DisplayName>Manufacturer</DisplayName>
  <Configuration>
    <VisibleBrowse>Show</VisibleBrowse>
    <VisibleEdit>Show</VisibleEdit>
    <VisibleNew>Show</VisibleNew>
  </Configuration>
</Field>
<Field name="ProductType" type="ShortText">
  <DisplayName>Type of the Product</DisplayName>
  <Configuration>
    <VisibleBrowse>Show</VisibleBrowse>
    <VisibleEdit>Show</VisibleEdit>
    <VisibleNew>Show</VisibleNew>
  </Configuration>
</Field>
<Field name="Description" type="LongText">
  <DisplayName>Description</DisplayName>
  <Configuration>
    <VisibleBrowse>Show</VisibleBrowse>
    <VisibleEdit>Show</VisibleEdit>
    <VisibleNew>Show</VisibleNew>
  </Configuration>
</Field>
<Field name="PublishDate" type="DateTime">
  <DisplayName>PublishDate</DisplayName>
  <Configuration>
    <VisibleBrowse>Show</VisibleBrowse>
    <VisibleEdit>Show</VisibleEdit>
    <VisibleNew>Show</VisibleNew>
  </Configuration>
</Field>
<Field name="Image" type="Image">
  <DisplayName>Cover image</DisplayName>
  <Description>Attached cover image. When persisted to the
repository it is handled as a reference to an image
content.</Description>
  <Bind property="ImageRef" />
  <Bind property="ImageData" />
  <Configuration>
    <VisibleBrowse>Show</VisibleBrowse>
    <VisibleEdit>Show</VisibleEdit>
    <VisibleNew>Show</VisibleNew>
    <ControlHint>sn:Image</ControlHint>
  </Configuration>
</Field>
</Fields>
</ContentType>

```

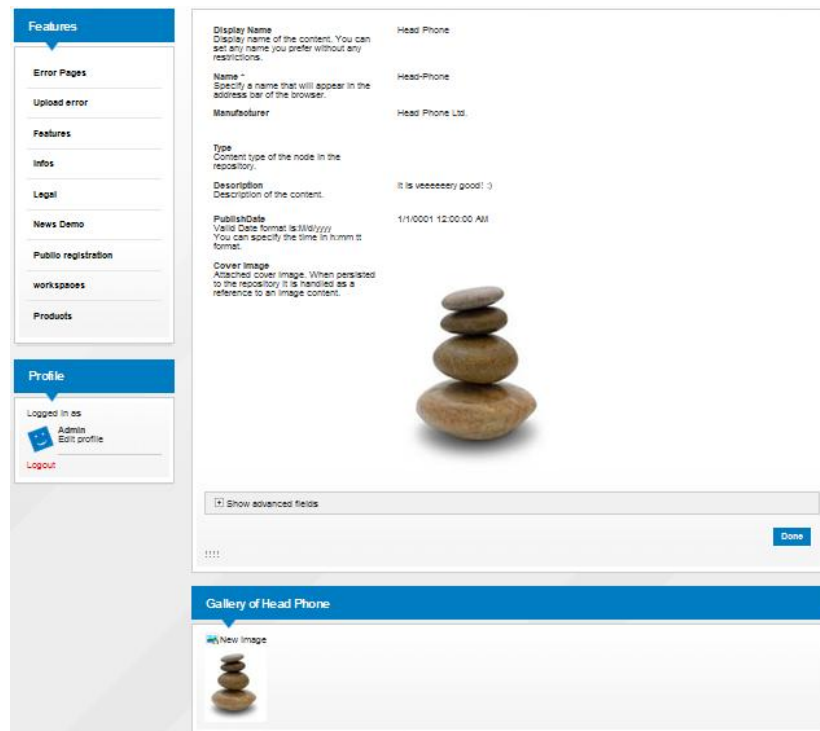
If everything goes well, you will have your new content type! If you go and create a new Product now somewhere in the portal and browse it, you'll find the end result exceptionally ugly – so don't worry. You're on the right track, it's just that this content type is handled as a Folder, which makes it look like a folder with child elements, lists, etc. We'll get there where it looks nice.

## 2.2 Define a content view

Like we mentioned, it looks quite ugly now but also promised to make it nicer. We're to use the content view for the built-in content type "Book" and polish it a bit.

1. First, let's create a Product! Create a folder for it under /Root/Sites/Default\_Site by clicking New and selecting Folder. You can name it anything (eg. Products).
2. Under this folder, create a new content (just click on New, select Product and enter some information – including a picture). If you click Browse now, you'll see it does not look fine.
3. Under /Root/Sites/Default\_Site/(apps) create a folder called "Product". Sense/Net uses name-based binding, meaning you need to have smart application folders with exactly the same name as your content type. Sense/Net will then locate that view/action for you and render your content according to that.
4. Copy the Browse action from /Root/Sites/Default\_Site/(apps)/Book to /Root/Sites/Default\_Site/(apps)/Product.
5. If you now browse your content, it will look a bit more like it. It does list the fields nicely and there's an extra portlet at the bottom of the page showing an image gallery for our product.

*Something like this:*



6. Now our content is rendered by the renderer of the Generic Content content type, located under /Root/Global/contentviews/Browse.ascx. If you enter some additional characters after the last </div>, you'll see them appearing in the portlet.
7. But we don't want ruin the browse action of the Generic Content, so let's create a new one! Create a folder called Product under /Root/Global/contentviews/ and copy /Root/Global/contentviews/Browse.ascx into it. Now if you modify this view, it'll only change the Browse view for the Product content type, not all the content.
8. Now open the ASCX file you just copied into your Product folder. Copy this into it, overriding those few lines in it currently:

```
<%@ Control Language="C#" AutoEventWireup="true"
Inherits="SenseNet.Portal.UI.SingleContentView" %>
<%@ Register Assembly="SenseNet.Portal"
Namespace="SenseNet.Portal.UI.Controls" TagPrefix="sn" %>

<div>
  <sn:ContextInfo runat="server" Selector="CurrentContext"
UsePortletContext="true"
  ID="myContext" />
  <div style="float: right">
    <asp:Panel ID="editPanel" runat="server" CssClass="book-rent">
```

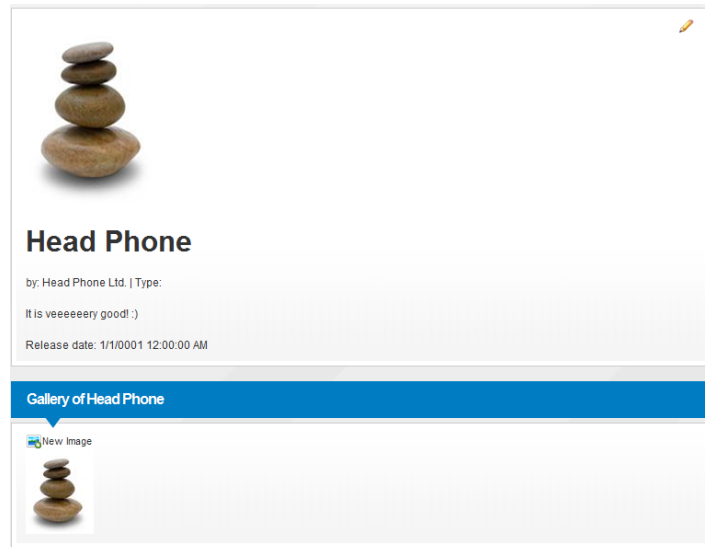
```

    <sn:actionlinkbutton ID="Actionlinkbutton2" runat="server"
    IconVisible="false" IconName="Edit"
    ImageUrl="/Root/skins/book/images/book_action_edit.png"
    actionname="Edit" contextinfoid="myContext" Text="Edit"
    style="padding: 10px;" title="Edit" />
    </asp:Panel>
  </div>
</div>
<div class="book-details">
  <div class="book-cover">
    <sn:Image CssClass="book-img" ID="Image" runat="server"
    FieldName="Image" RenderMode="Browse">
      <browsetemplate>
        <asp:Image CssClass="book-img"
        ImageUrl="/Root/Global/images/missingcover.png" Width="150"
        Height="200" ID="ImageControl" runat="server" alt="" />
      </browsetemplate>
    </sn:Image>
  </div>
  <div class="book-header-container">
    <div class="book-header">
      <div class="book-title">
        <h1>
          <%= GetValue("DisplayName") %>
        </h1>
      </div>
      <div class="book-author">
        by: <%= GetValue("Vendor") %> | Type: <%=
    GetValue("Type") %>
      </div>

      <div class="book-summary">
        <%= GetValue("Description") %>
      </div>
      <asp:Panel ID="futureReleasePanel" runat="server"
    CssClass="book-rent">
        Release date:
        <%=GetValue("PublishDate") %>
      </asp:Panel>
    </div>
  </div>
</div>

```

We are borrowing some stylesheet elements from the Book CSS to make it look nicer. Now if you open your product's browse view, it looks like this:



Did you notice the little pencil at the top right corner? I'll let you figure out how it got there... ☺

It's time to explain how the gallery portlet got there. We copied the Browse action of the "Book" content type – remember? The browse view of that particular content type features two portlets: the one displaying it and one more, the gallery portlet. If you locate the Browse action (/Root/Sites/Default\_Site/(apps)/Product/Browse) and hit "Browse", then open the Portal Remote Control and hit edit, you'll see the Application Page. Anything you modify here will be reflected in every Browse view of the Product content type. And remember, Browse view is the one which is displayed to normal users when opening the product.

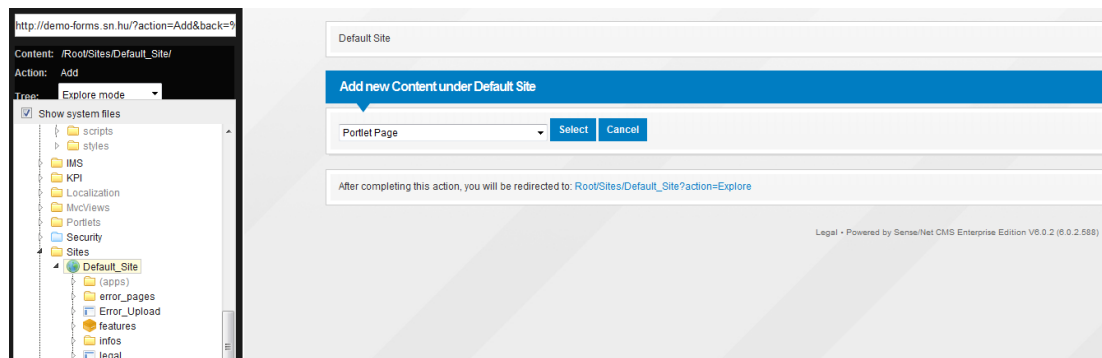
Let's say this looks nice and move on!

### 2.3 Define the basic structure

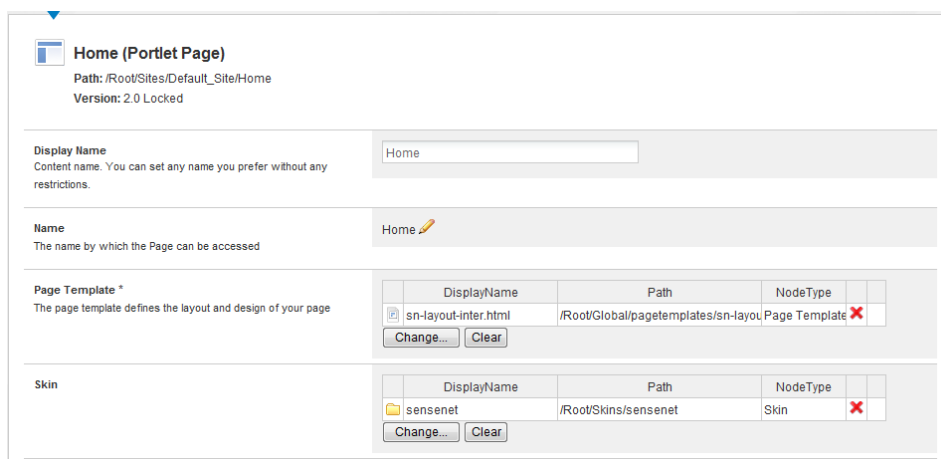
Okay, we have a content type and it is displayed "nicely". Now let's define a basic structure!

Let's say we're to have an opening page with some welcome text and the latest 3 products. We'll have a sub-page which lists all the products plus a product page which will be opened when you click on a product. (Note: the last one we just created, remember? You don't need a page for every product, Sense/Net renders it for you for every content in that type.)

1. Under /Root/Sites/Default Site/ create a new Portlet Page.



- Fill the new page form. Name you page for example as "Home", select a Page Template (sn-layout-inter.html) and a Skin (sensenet). The rest you can leave empty.



DisplayName	Path	NodeType
sn-layout-inter.html	/Root/Global/pagetemplates/sn-layout	Page Template

DisplayName	Path	NodeType
sensenet	/Root/Skins/sensenet	Skin

- If you open <http://<<yourSite>>/home>, you'll see your empty page with a navigation portlet. The navigation portlet is there because it is on the page template. You can play around and select a page template you like better – we'll use this for the evaluation.
- Let's create a second page called "Products" under Home. I'll leave this to you, you should be able to do this based on the previous steps.

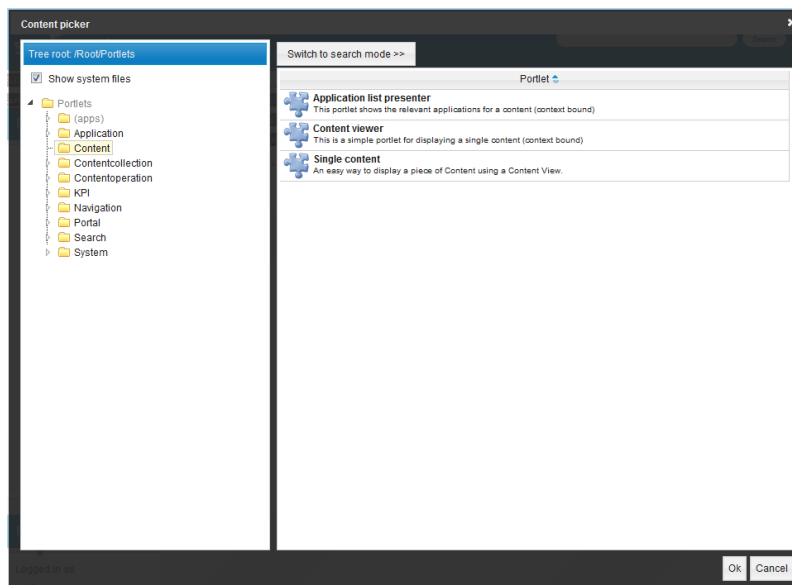
## 2.4 Put portlets on the page

Let's populate our pages! First Home, then Products. You can add portlets to a page by opening the Portal Remote Control, switching to edit mode and then clicking on the "Add portlet" link in the desired zone. Portlets can be rearranged by drag-and-drop, while their properties can be opened by clicking on the little down arrow on their top-right and selecting Edit.



## 2.4.1 Simple static content presentation

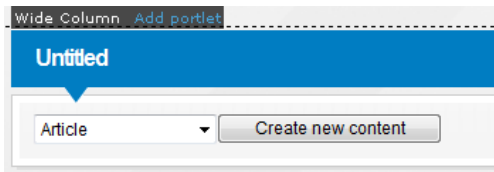
1. Let's add the Single Content Portlet to the Wide Column.



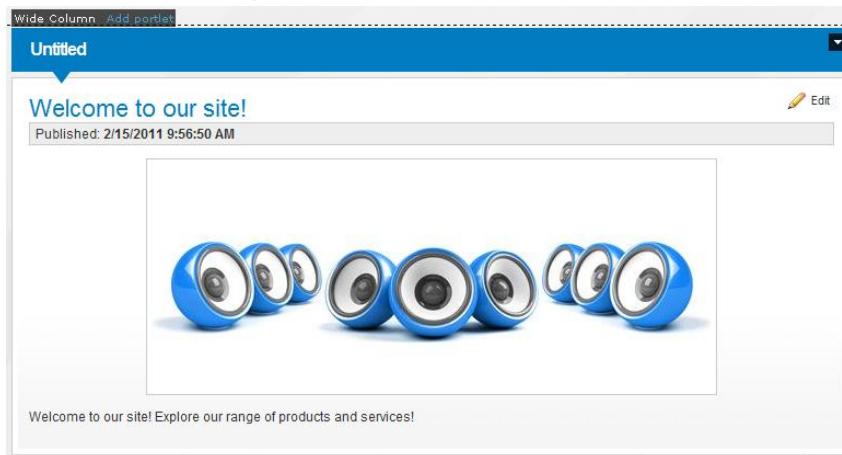
2. Open the portlet property drop-down and click on New Content.



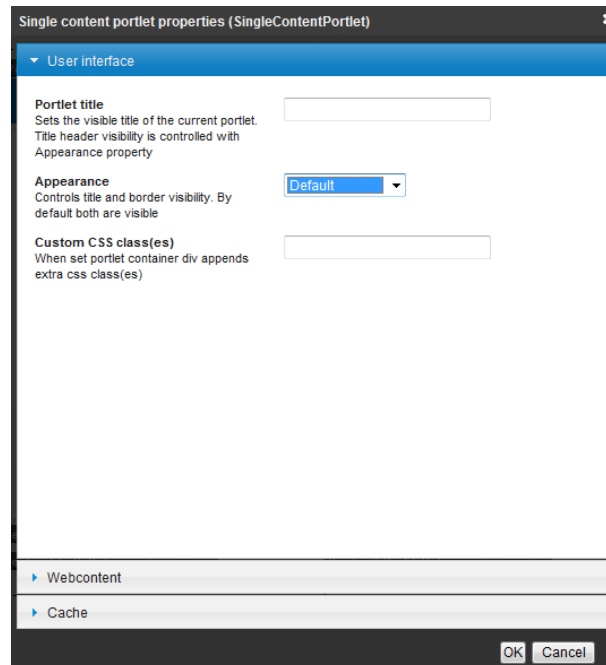
3. You can play around with the three available content types and explore them all, we'll use the Article for this evaluation guide.



4. Create some nice content, and save it.

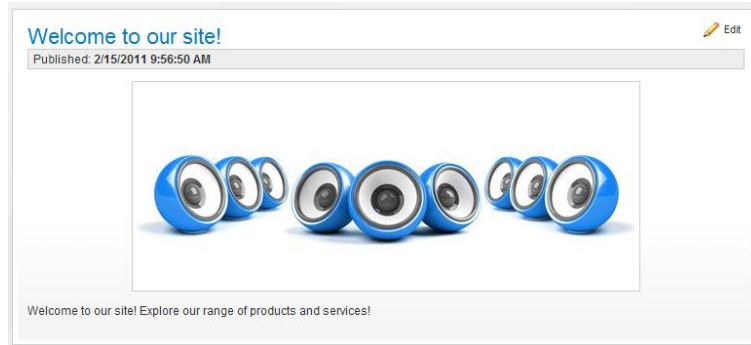


5. Open the portlet's properties by opening the drop-down at the top-right corner of the portlet and clicking on Edit (NOT Edit Content, which edits the content!!!).



6. If you want your portlet to have a portlet header, type some portlet title and explore the different Appearance options. We'll select "Border Only".
7. To save the page, open the Portal Remote Control and click on Check-in. Please note that Sense/Net features check-out/check-in and versioning for

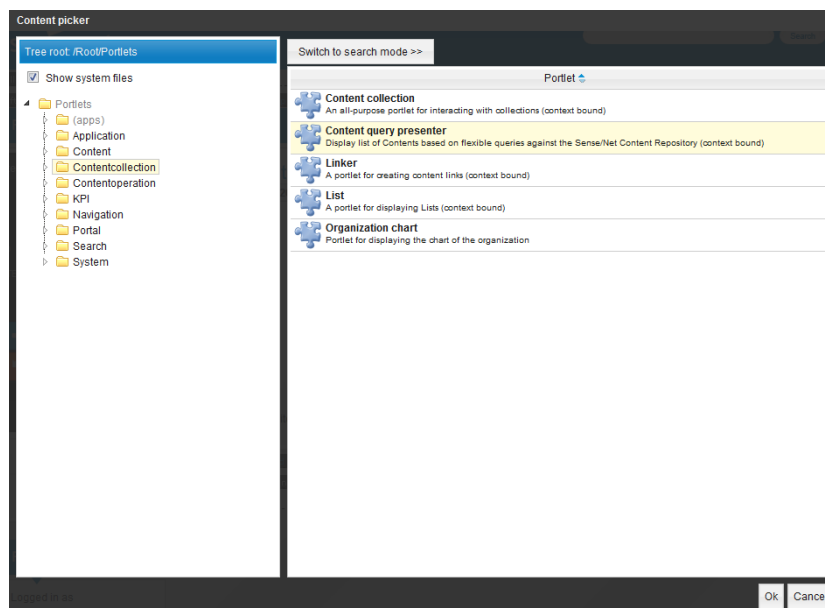
every content, including pages. Always check in a page after finished editing it, so other users may edit it as well.



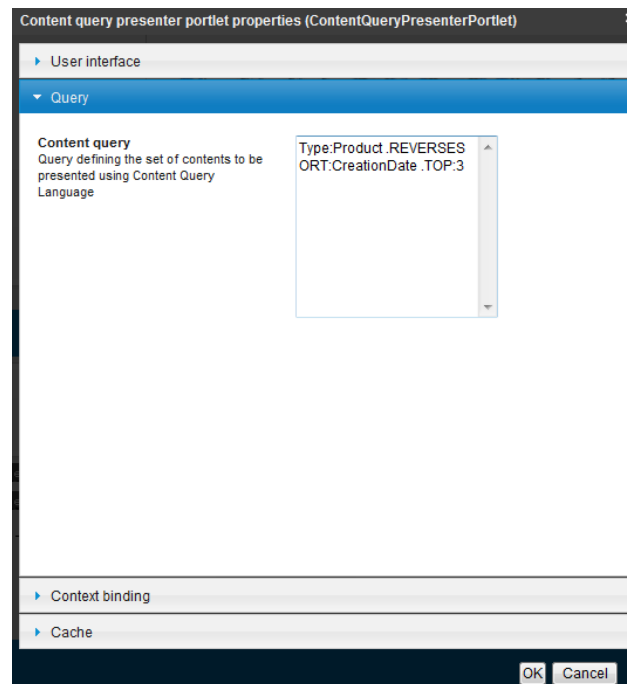
## 2.4.2 Query

First, we'll insert the latest 3 products portlet on the Home page, then the all product listings on the product page.

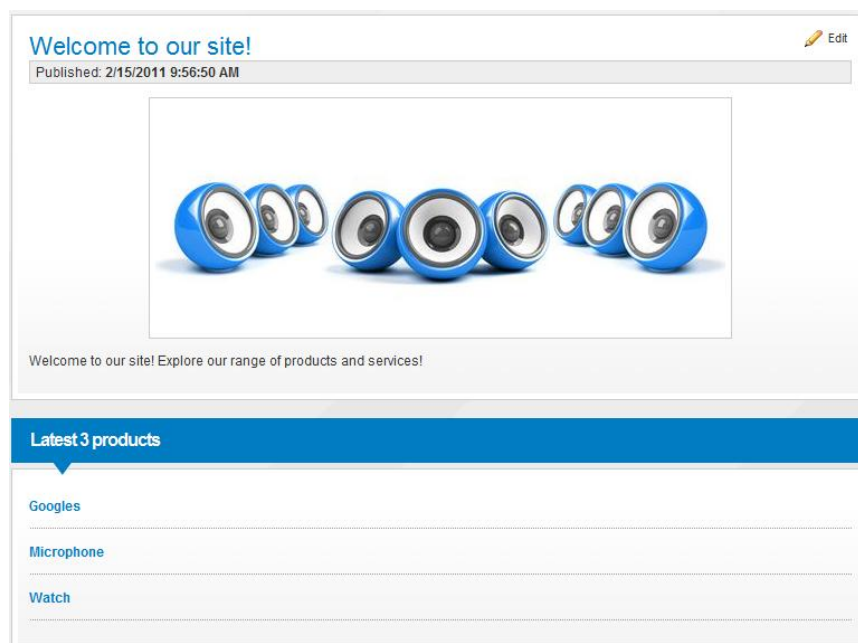
1. Open the Home page for edit again and insert the Content Query Presenter portlet to the Wide Column.



2. Drag it to be under the static content portlet!
3. Open its properties for edit. Let's have a title like "Latest 3 products" and select "Title and Border" for appearance.
4. Insert the query into the Content Query box: Type:Product  
.REVERSESORT:CreationDate .TOP:3



5. Save the portlet then check-in the page.
6. Don't forget to upload some product to see the results!



Now if you do not like the rather simple presentation of the list, you may create a new ASCX showing a different set of fields, eg. the vendor, image, etc. To do this you need to create an ASCX somewhere in the hierarchy (I'd suggest /Root/Global/renderers/) and the point to that ASCX from the portlet's Renderer setting.

You can try this, for example, which writes the vendor under the name of the product.

```

<%@ Control Language="C#" AutoEventWireup="true" %>

<asp:ListView ID="ContentList" runat="server"
  EnableViewState="false">
  <LayoutTemplate>
    <div>
      <asp:PlaceHolder ID="itemPlaceHolder" runat="server" />
    </div>
  </LayoutTemplate>
  <ItemTemplate>
    <div class="sn-content sn-contentlist-item"><h1 class="sn-
  content-title">
      <sn:ActionLinkButton NodePath='<%=Eval("Path") %>'
        ActionName="Browse"
        IconVisible="false" runat="server">
        <%=Eval("DisplayName") %>
      </sn:ActionLinkButton></h1>
      By:<%=Eval("Vendor") %>
    </div>
  </ItemTemplate>
</asp:ListView>
  
```

Now the Products page:

1. Go to the Products page (/Home/Products), open the Portal Remote Control and switch to edit mode.
2. Add the Content Query Presenter portlet to the Wide Column.
3. The query should be Type:Product .SORT:Name this time, as we'll list the products alphabetically.
4. Save the portlet, check in the page.

*Now, we want to do some nicer look-and-feel, so we'll do a trick: we're going to create 2 ASCXs, one to display a single Product as a list element, then another which calls this one as many times as needed.*

5. Create a new ProductListings.ascx under /Root/Global/renderers/ as we'll want some extra fields to appear in the portlet. The ASCX should contain this:

```

<%@ Control Language="C#" AutoEventWireup="true" %>
<%@ Register TagPrefix="sn" Assembly="SenseNet.CorePortlets"
  Namespace="SenseNet.Portal.Portlets" %>
  
```

```

<asp:ListView ID="ContentList" runat="server"
EnableViewState="false">
  <LayoutTemplate>
    <div>
      <asp:Placeholder ID="itemPlaceholder" runat="server" />
    </div>
  </LayoutTemplate>
  <ItemTemplate>
    <div>
      <sn:ContentPortlet id="content" runat="server"
BindTarget="CustomRoot" CustomRootPath='<%=Eval("Path")%>'
ViewPath="/Root/Global/contentviews/Product/ListItem.ascx">
      </sn:ContentPortlet>
    </div>
  </ItemTemplate>
</asp:ListView>

```

6. Create a ListItem.ascx under /Root/Global/contentviews/Product and copy this into it:

```

<%@ Control Language="C#" AutoEventWireup="true"
Inherits="SenseNet.Portal.UI.SingleContentView" %>

<script runat="server">
  protected string GetParentTitle(object cnt)
  {
    return
    ((SenseNet.ContentRepository.Content)cnt).ContentHandler.Parent.GetPr
operty<string>("Title");
  }
  protected string GetAction(string action)
  {
    return
    SenseNet.ApplicationModel.ActionFramework.GetActionUrl(this.Content.C
ontentHandler.Path, action,
    SenseNet.Portal.Virtualization.PortalContext.Current.ContextNodePath)
;
  }
}

</script>

<div class="book-details">
<table cellpadding=5><tr><td>
  <div class="book_cover">
    <a href="<%=GetAction("Browse") %>">
      <sn:image id="Image" runat="server" fieldname="Image"
rendermode="Browse">
      <BrowseTemplate>

```

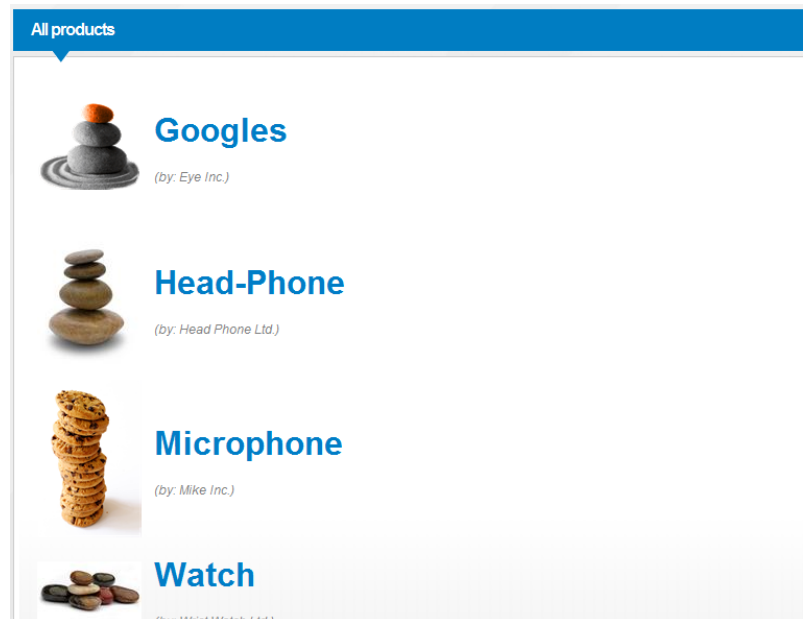
```

      <asp:Image
ImageUrl="/Root/Global/images/missingcover.png" Width="100"
ID="ImageControl" runat="server" alt="" />
      </BrowseTemplate>
    </sn:image>
  </a>
</div>
</td><td>
<div class="book-header-container">
  <div class="book-header">
    <div class="book-title">
      <h1>
        <a href="<%=GetAction("Browse") %>">
          <%= GetValue("Name") %>
        </a>
      </h1>
    </div>
    <div class="book-mainpage">
    </div>
    <div class="book-author">
      <span style="color: gray; font-style: italic;">(by: <%=
GetValue("Vendor") %>) </span>
    </div>
  </div>
</div>
</td></tr></table>
</div>

```

(Yes, we are using the Book CSS again.)

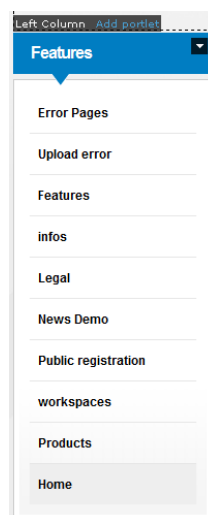
- Open your query portlet at page Products and set the new ProductListings.ascx as renderer. The end result should look like this:



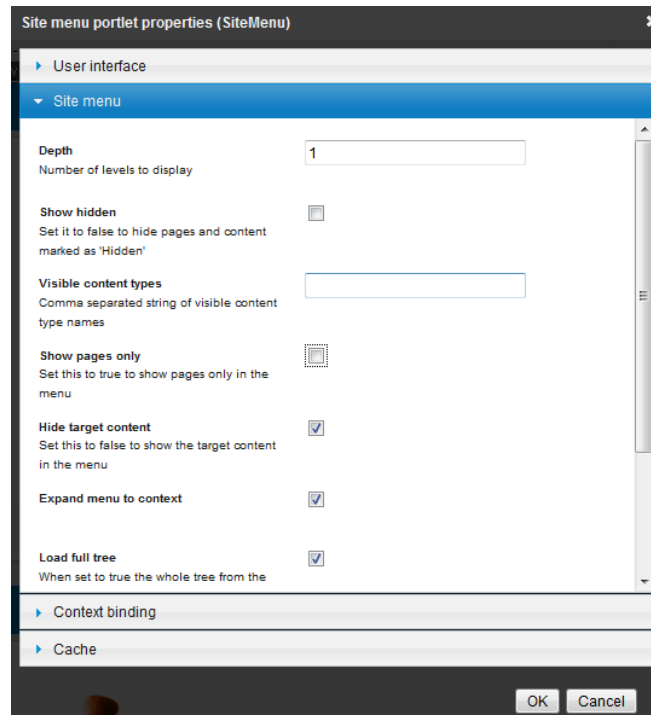
Let's assume this is enough. You may play around and modify the CSS under /Root/Skins or refine the ASCX more if you like.

## 2.5 A word on navigation

Navigation is also a query in Sense/Net. It queries certain content and display them in a narrow portlet. The navigation portlet on the site queries every content type and displays them under each other, ordered by their Index value.



You can replace this or just modify its properties to display only content with type Page, and so on. Just switch to edit mode and open the properties of the portlet (top-right arrow) and play with the settings a bit.



## 2.6 Some basic design stuff

- The page template we used in this guide is here:  
`/Root/Global/pagetemplates/sn-layout-inter.html`
- The Book CSS we borrowed for our ASCXs is here:  
`/Root/Skins/book/styles/book.css`
- The blue skin CSS is here: `/Root/Skins/sensenet/styles/skin.css`

Go ahead and explore them by using some page inspection tool – that way you'll learn a lot more than reading guides. Start by modifying the classes we use in the product listings representation.

Explore the Wiki for more information on the skin system.