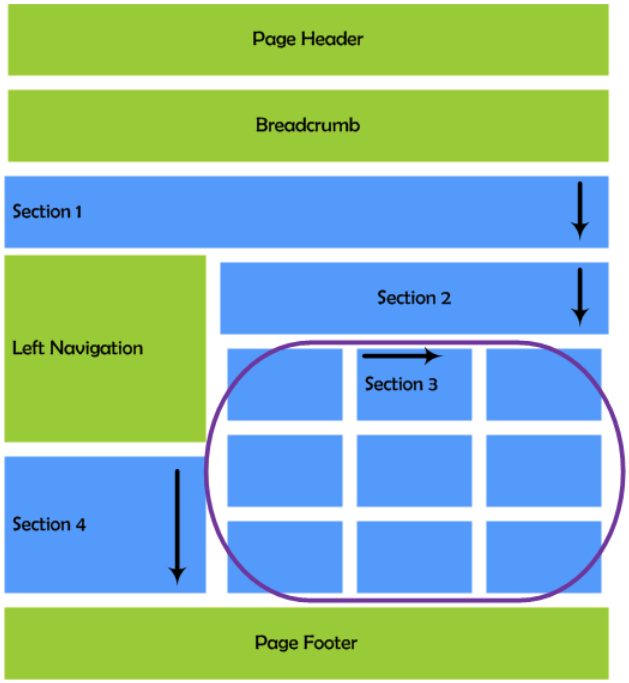
Creating Contepages

Category Page Template

The Category Page Template displays the Facet Navigation, merchandised content from the hybris WCMS Cockpit, but does not show the product set. To show the product set as well as merchandised content, an alternative template from the Product Catalog layout types should be used.



In hybris divide content pages are divided into two types :

- **Static Content pages.**
- **Dynamic Content Pages.**

Static content pages :
Which can be created by using impex directly no need to any jsp or controller.. default controller and default jsp can easily handle that.

Dynamic content pages: for this type we will need our own controller and jsp.

1)StaticContentpage

Let us create a static content page.

Step 1: First requirement for creating any page in hybris is layout i.e. template.
As we have so many predefined layouts which we can use to create our new content pages.
For ex. landingLayout1,landingLayout2, landingLayout3 etc.

INSERT_UPDATE PageTemplate;$contentCV[unique=true];uid[unique=true];name; frontendTemplateName; restrictedPageTypes(code);active[default=true]
;;LandingPage1Template;LandingPage1Template;layout/landingLayout1Page;CategoryPage, ContentPage

Now, our Template page is ready for use..

Step 2: Next thing needed is to create the content page by using this template.
(for demonstation we will create Privacy Policy page of thebodyshop site).

INSERT_UPDATE ContentPage; $contentCV[unique=true];uid[unique=true] ;name; masterTemplate(uid,$contentCV);label;defaultPage[default='true'];approvalStatus(code) [default='approved'];homepage[default='false']
; ;NewsPage ;News Page ;ContentPage1Template;/news

Note: Here label field plays the important role as we have to access this page based on this level. for e.x. to access this page we have to type : http://thebodyshop.local:9001/thebodyshopstorefront/news
Now our content page is ready.

Step 3: Now, we can Enter the content to this content page.. we can directly use CMSParagraph component for this purpose. even we can enter any number of component as per the requirement.

INSERT_UPDATE CMSParagraphComponent;$contentCV[unique=true];uid[unique=true] ;name ;content[lang=$lang] ;&componentRef
; ;NewspageContent ; News page Content ; Hai, Techouts nominated in For the Award:::::: ;NewspageContent

Step 4: Now next step is to assign this Component to the slot :

INSERT_UPDATE ContentSlot;$contentCV[unique=true];uid[unique=true];name ; active ; cmsComponents(&componentRef)
; ;NewsPageSlot ;News PageSlot ;true ;NewspageContent

Note: Here if you see (it is cmsComponents) it can take any number of component separated by comma (,)

Step 5: Last step is to assign this slot to the page.

INSERT_UPDATE ContentSlotForPage;$contentCV[unique=true];uid[unique=true] ;position[unique=true];page(uid,$contentCV)[unique=true];contentSlot(uid,$contentCV)[unique=true]				
;	;NewsPageSlotpage	;Section1	;NewsPage	; NewsPageSlot

Note: here position is Section so you have to see the section available for the template being used.

Summary:

1) Create the layout i.e template.
2) By using Layout(template) create the content page.
3) Enter the content to the content page.
4) assign this Component (content Page) to the slot.
5) assign the slot to the page.

Task 1:Cteate Staticcontentpage for Newslinkcomponetnt

```
#
# Import the CMS content for the B2C Telco site
#
$contentCatalog=b2ctelcoContentCatalog
$contentCV=catalogVersion(CatalogVersion.catalog(Catalog.id[default=$contentCatalog]),CatalogVersion.version[default=Staged])[default=$contentCatalog:Staged]

# Import modulegen config properties into impex macros
UPDATE GenericItem[processor=de.hybris.platform.commerceservices.impex.impl.ConfigPropertyImportProcessor];pk[unique=true]
$jarResourceCms=$config-jarResourceCmsValue
$jarResourceCmsTelco=jar:de.hybris.platform.b2ctelcostore.setup.B2ctelcoStoreSystemSetup&/b2ctelcostore/import/cockpits/cmscockpit
$addonExtensionName=b2ctelcostorefront
$lang=en


INSERT_UPDATE ContentPage; $contentCV[unique=true];uid[unique=true] ;name; masterTemplate(uid, $contentCV);label;defaultPage[default='true'];approvalStatus(code)
```

```
[default='approved'];homepage[default='false']
                ;                ;NewsPage        ;News Page      ;ContentPage1Template;/news


INSERT_UPDATE CMSParagraphComponent;$contentCV[unique=true];uid[unique=true] ;name
;content[lang=$lang]            ;&componentRef
                        ;                ;NewspageContent  ; News page Content   ; Hai, Techouts nominated in
For the Award:::::: ;NewspageContent


INSERT_UPDATE ContentSlot;$contentCV[unique=true];uid[unique=true];name          ; active   ;
cmsComponents(&componentRef)
                ;                ;NewsPageSlot     ;News PageSlot ;true      ;NewspageContent

INSERT_UPDATE ContentSlotForPage;$contentCV[unique=true];uid[unique=true]
;position[unique=true];page(uid,$contentCV)[unique=true];contentSlot(uid,$contentCV)[unique=true]
                ;                ;NewsPageSlotpage ;Section1          ;NewsPage                ;
NewsPageSlot
```

After Synchronise click on **news link** then it will display **Hai, Techouts nominated in For the Award:::::: ;NewspageContent**

2) Dynamic Content Page:

For Creating Dynamic Content Page we need to create jsp and Controller Separately.

Steps to Creating Dynamic Content Page:

Step1:-

Create the impex file as mentioned above in Static Content page and while creating content page no need to give content through impex. For giving the Dynamic Content we need to work with Custom Controller and Jsp files.

Step 2:-

Creating the controller

While creating the controller we have to give RequestMapping value as the label name. And write the logic and return to jsp page

Step3

Create the jsp in the specified location to display the required content .While displaying the jsp page we can work with default template like.

```
<template:page pageTitle="${pageTitle}">
<!--……..
……..--></template:page>
```

Task2:Create DynamicContent page it will display firstname and last name of customer

1.Create impex for dynamic contentpage like this

```
#
# Import the CMS content for the B2C Telco site
#
$contentCatalog=b2ctelcoContentCatalog
$contentCV=catalogVersion(CatalogVersion.catalog(Catalog.id[default=$contentCatalog]),CatalogVersion.version[default=Staged])[default=$contentCatalog:Staged]

# Import modulegen config properties into impex macros
UPDATE GenericItem[processor=de.hybris.platform.commerceservices.impex.impl.ConfigPropertyImportProcessor];pk[unique=true]
$jarResourceCms=$config-jarResourceCmsValue
$jarResourceCmsTelco=jar:de.hybris.platform.b2ctelcostore.setup.B2ctelcoStoreSystemSetup&/b2ctelcostore/import/cockpits/cmscockpit
$addonExtensionName=b2ctelcostorefront
$lang=en


INSERT_UPDATE ContentPage; $contentCV[unique=true];uid[unique=true] ;name; masterTemplate(uid,$contentCV);label;defaultPage[default='true'];approvalStatus(code)[default='approved'];homepage[default='false']

                ;                    ;TermsandconditionsPage        ;Termsandconditions Page ;ContentPage1Template;/Termsandconditions

INSERT_UPDATE ContentSlot;$contentCV[unique=true];uid[unique=true];name          ; active   ; cmsComponents(&componentRef)
                    ;                    ;TermsandconditionsPageSlot     ;Termsandconditions Page Slot ;true ;;
INSERT_UPDATE ContentSlotForPage;$contentCV[unique=true];uid[unique=true] ;position[unique=true];page(uid,$contentCV)[unique=true];contentSlot(uid,$contentCV)[unique=true]

                        ;                    ;TermsandconditionsSlotpage ;Section1  ;TermsandconditionsPage ; ;
```

2.Create the controller

```
/**
 *
 */
package com.lycamobile.storefront.controllers.pages;
```

```
import
de.hybris.platform.acceleratorstorefrontcommons.breadcrumb.ResourceBreadcrumbBuilder;
import
de.hybris.platform.acceleratorstorefrontcommons.controllers.pages.AbstractPageController;
import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.commercefacades.customer.CustomerFacade;
import de.hybris.platform.commercefacades.user.data.CustomerData;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;


/**
 * @author jagadish
 *
 */
@Controller
@RequestMapping(value = "/Termsandconditions")
public class TermsAndConditionsController extends AbstractPageController
{

	@Resource(name = "customerFacade")
	protected CustomerFacade customerFacade;
	@Resource(name = "accountBreadcrumbBuilder")
	private ResourceBreadcrumbBuilder accountBreadcrumbBuilder;
	private static final String TERMSANDCONDITIONS_PAGE =
"TermsandconditionsPage";

	@RequestMapping(method = RequestMethod.GET)
	public String termsAndConditions(final Model model) throws
CMSItemNotFoundException
	{

		final CustomerData customerData = customerFacade.getCurrentCustomer();


		model.addAttribute("customerData", customerData);

		storeCmsPageInModel(model,
getContentPageForLabelOrId(TERMSANDCONDITIONS_PAGE));
```

```
                setUpMetaDataForContentPage(model,
getContentPageForLabelOrId(TERMSANDCONDITIONS_PAGE));
                model.addAttribute("breadcrumbs",
accountBreadcrumbBuilder.getBreadcrumbs("text.account.profile"));
                model.addAttribute("metaRobots", "noindex,nofollow");
                return "pages/account/termsandcondions";
        }


}
```

Note: Here, **/Termsandconditions** is the label name of the news Content Page. When request is coming this Controller is executed. And returns one jsp.Here, its returns **termsandconditions.jsp**

Now Create the jsp page like this

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
 pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<%@ page trimDirectiveWhitespaces="true" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<%@ taglib prefix="template" tagdir="/WEB-INF/tags/desktop/template" %>
<%@ taglib prefix="common" tagdir="/WEB-INF/tags/desktop/common" %>


      <div id="globalMessages">
            <common:globalMessages/>
      </div>



<template:page pageTitle="${pageTitle}">

      Welcome Mr... <br> ${data.firstName}   ${data.lastName}


</template:page>
```