# Product Detail页面显示逻辑

// 用来显示某个product详情 或者编辑购物车中product页面

ProductController.ProductDetails(int productId, int updatecartitemid = 0)

## FeatureedProducts

IPagedList<Product> featuredProducts = null;

featuredProducts = \_productService.SearchProducts(

categoryIds: new List<int> { category.Id },

storeId: \_storeContext.CurrentStore.Id,

visibleIndividuallyOnly: true,

featuredProducts: true);

hasFeaturedProductsCache = featuredProducts.TotalCount > 0;

## Product attributes

**dbo.ProductAttribute**： Custom TextColor 这些属性

**dbo.Product\_ProductAttribute\_Mapping**： ProductId, AttributeId, 及属性的要求（IsRequired, AttributeControlType）

**dbo.ProductAttributeValue**: AttributeControlType不是TextBox MultilineTextbox Datepicker FileUpload时**ProductAttributeMappingId**对应的选项值

IList<ProductAttributeMapping> productAttributeMapping = \_productAttributeService.GetProductAttributeMappingsByProductId(product.Id);

public virtual IList<ProductAttributeMapping> GetProductAttributeMappingsByProductId(int productId)

{

string key = string.Format(PRODUCTATTRIBUTEMAPPINGS\_ALL\_KEY, productId);

return \_cacheManager.Get(key, () =>

{

var query = from pam in \_productAttributeMappingRepository.Table

orderby pam.DisplayOrder

where pam.ProductId == productId

select pam;

var productAttributeMappings = query.ToList();

return productAttributeMappings;

});

}

// AttributeControlType为TextBox MultilineTextbox Datepicker FileUpload 不需要值初始化

if (attribute.ShouldHaveValues())

{

//values

var attributeValues = \_productAttributeService.**GetProductAttributeValues**(attribute.Id);

foreach (var attributeValue in attributeValues)

{

var valueModel = new ProductDetailsModel.ProductAttributeValueModel

{

Id = attributeValue.Id,

Name = attributeValue.GetLocalized(x => x.Name),

ColorSquaresRgb = attributeValue.ColorSquaresRgb, //used with "Color squares" attribute type

IsPreSelected = attributeValue.IsPreSelected

};

attributeModel.Values.Add(valueModel);

//display price if allowed

if (\_permissionService.Authorize(StandardPermissionProvider.DisplayPrices))

{

decimal taxRate;

decimal attributeValuePriceAdjustment = \_priceCalculationService.GetProductAttributeValuePriceAdjustment(attributeValue);

decimal priceAdjustmentBase = \_taxService.GetProductPrice(product, attributeValuePriceAdjustment, out taxRate);

decimal priceAdjustment = \_currencyService.ConvertFromPrimaryStoreCurrency(priceAdjustmentBase, \_workContext.WorkingCurrency);

if (priceAdjustmentBase > decimal.Zero)

valueModel.PriceAdjustment = "+" + \_priceFormatter.FormatPrice(priceAdjustment, false, false);

else if (priceAdjustmentBase < decimal.Zero)

valueModel.PriceAdjustment = "-" + \_priceFormatter.FormatPrice(-priceAdjustment, false, false);

valueModel.PriceAdjustmentValue = priceAdjustment;

}

//"image square" picture (with with "image squares" attribute type only)

if (attributeValue.ImageSquaresPictureId > 0)

{

var productAttributeImageSquarePictureCacheKey = string.Format(ModelCacheEventConsumer.PRODUCTATTRIBUTE\_IMAGESQUARE\_PICTURE\_MODEL\_KEY,

attributeValue.ImageSquaresPictureId,

\_webHelper.IsCurrentConnectionSecured(),

\_storeContext.CurrentStore.Id);

valueModel.ImageSquaresPictureModel = \_cacheManager.Get(productAttributeImageSquarePictureCacheKey, () =>

{

var imageSquaresPicture = \_pictureService.GetPictureById(attributeValue.ImageSquaresPictureId);

if (imageSquaresPicture != null)

{

return new PictureModel

{

FullSizeImageUrl = \_pictureService.GetPictureUrl(imageSquaresPicture),

ImageUrl = \_pictureService.GetPictureUrl(imageSquaresPicture, \_mediaSettings.ImageSquarePictureSize)

};

}

return new PictureModel();

});

}

//picture of a product attribute value

valueModel.PictureId = attributeValue.PictureId;

}  
  
}

public virtual IList<ProductAttributeValue> GetProductAttributeValues(int productAttributeMappingId)

{

string key = string.Format(PRODUCTATTRIBUTEVALUES\_ALL\_KEY, productAttributeMappingId);

return \_cacheManager.Get(key, () =>

{

var query = from pav in \_productAttributeValueRepository.Table

orderby pav.DisplayOrder

where pav.ProductAttributeMappingId == productAttributeMappingId

select pav;

var productAttributeValues = query.ToList();

return productAttributeValues;

});

}

# Category 下的Product

//products

IList<int> alreadyFilteredSpecOptionIds = model.PagingFilteringContext.SpecificationFilter.GetAlreadyFilteredSpecOptionIds(\_webHelper);

IList<int> filterableSpecificationAttributeOptionIds;

var products = \_productService.SearchProducts(out filterableSpecificationAttributeOptionIds,

true,

categoryIds: categoryIds,

storeId: \_storeContext.CurrentStore.Id,

visibleIndividuallyOnly: true,

featuredProducts: \_catalogSettings.IncludeFeaturedProductsInNormalLists ? null : (bool?)false,

priceMin: minPriceConverted,

priceMax: maxPriceConverted,

filteredSpecs: alreadyFilteredSpecOptionIds,

orderBy: (ProductSortingEnum)command.OrderBy,

pageIndex: command.PageNumber - 1,

pageSize: command.PageSize);

model.Products = PrepareProductOverviewModels(products).ToList();

//specs

model.PagingFilteringContext.SpecificationFilter.PrepareSpecsFilters(alreadyFilteredSpecOptionIds,

filterableSpecificationAttributeOptionIds != null ? filterableSpecificationAttributeOptionIds.ToArray() : null,

\_specificationAttributeService,

\_webHelper,

\_workContext,

\_cacheManager);