decimal minimumCustomerEnteredPrice = \_currencyService.**ConvertFromPrimaryStoreCurrency**(product.MinimumCustomerEnteredPrice, \_workContext.WorkingCurrency);

decimal maximumCustomerEnteredPrice = \_currencyService.**ConvertFromPrimaryStoreCurrency**(product.MaximumCustomerEnteredPrice, \_workContext.WorkingCurrency);

/// currencySettings.PrimaryStoreCurrencyId安装后值是1, en-US的currency

///

public virtual decimal **ConvertFromPrimaryStoreCurrency**(decimal amount, Currency targetCurrencyCode)

{

var primaryStoreCurrency = GetCurrencyById(\_currencySettings.PrimaryStoreCurrencyId);

var result = **ConvertCurrency**(amount, primaryStoreCurrency, targetCurrencyCode);

return result;

}

/// <summary>

/// Converts currency source和target不一样再实行转换

/// 因为currency表保存的Rate都是相对于PrimaryCurrency的， 所以需要先转换为primary的值然后转换为target

/// ConvertToPrimaryExchangeRateCurrency 先转换为PrimaryCurrency

/// ConvertFromPrimaryExchangeRateCurrency PrimaryCurrency转换为target

/// </summary>

public virtual decimal ConvertCurrency(decimal amount, Currency sourceCurrencyCode, Currency targetCurrencyCode)

{

if (sourceCurrencyCode == null)

throw new ArgumentNullException("sourceCurrencyCode");

if (targetCurrencyCode == null)

throw new ArgumentNullException("targetCurrencyCode");

decimal result = amount;

if (sourceCurrencyCode.Id == targetCurrencyCode.Id)

return result;

if (result != decimal.Zero && sourceCurrencyCode.Id != targetCurrencyCode.Id)

{

result = ConvertToPrimaryExchangeRateCurrency(result, sourceCurrencyCode);

result = ConvertFromPrimaryExchangeRateCurrency(result, targetCurrencyCode);

}

return result;

}

## \_priceFormatter

\_priceFormatter.FormatPrice(minimumCustomerEnteredPrice, false, false)

Libraries\Nop.Services\Catalog\PriceFormatter.cs

public virtual string FormatPrice(decimal price, bool showCurrency, bool showTax)

{

bool priceIncludesTax = \_workContext.TaxDisplayType == TaxDisplayType.IncludingTax;

return FormatPrice(price, showCurrency, \_workContext.WorkingCurrency, \_workContext.WorkingLanguage, priceIncludesTax, showTax);

}

public virtual string FormatPrice(decimal price, bool showCurrency,

Currency targetCurrency, Language language, bool priceIncludesTax, bool showTax)

{

//we should round it no matter of "ShoppingCartSettings.RoundPricesDuringCalculation" setting

price = RoundingHelper.RoundPrice(price);

string currencyString = GetCurrencyString(price, showCurrency, targetCurrency);

if (showTax)

{

//show tax suffix

string formatStr;

if (priceIncludesTax)

{

formatStr = \_localizationService.GetResource("Products.InclTaxSuffix", language.Id, false);

if (String.IsNullOrEmpty(formatStr))

formatStr = "{0} incl tax";

}

else

{

formatStr = \_localizationService.GetResource("Products.ExclTaxSuffix", language.Id, false);

if (String.IsNullOrEmpty(formatStr))

formatStr = "{0} excl tax";

}

return string.Format(formatStr, currencyString);

}

return currencyString;

}

ProductDetail （不是这个文档的内容）

## 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;

});

}