

Document Version 1.3

<b>Current Page</b> @Model	<b>Collections</b> .Children .[DocTypeName]s (Pluralized) .Ancestors() .Ancestors(int level) .Ancestors(string nodeTypeAlias) .Ancestors(Func< DynamicNode, bool > func) .AncestorsOrSelf() .AncestorsOrSelf(string nodeTypeAlias) .AncestorsOrSelf(int level) .AncestorsOrSelf(Func< DynamicNode, bool > func) .Descendants() .Descendants(string nodeTypeAlias) .Descendants(int level) .Descendants(Func< INode, bool > func) .DescendantsOrSelf() .DescendantsOrSelf(int level) .DescendantsOrSelf(string nodeTypeAlias) .DescendantsOrSelf(Func< INode, bool > func) .XPath(string xPath) .GetChildrenAsList	<b>Traversing</b> .Parent .First() .Last() .Up() .Up(int) (note 0 = 1) .Down() .Down(int) (note 0 = 1) .Next() .Next(int) (note 0 = 1) .Previous() .Previous(int) (note 0 = 1) .AncestorOrSelf() .AncestorOrSelf(string nodeTypeAlias) .AncestorOrSelf(int level) .AncestorOrSelf(Func< DynamicNode, bool > func) .DescendantOrSelf() .DescendantOrSelf(string nodeTypeAlias) .DescendantOrSelf(int level) .DescendantOrSelf(Func< DynamicNode, bool > func)	<b>Property Checking</b> .HasProperty(string propertyAlias) .HasValue(string propertyAlias) .IsNull(string propertyAlias)  <b>IsHelpers</b> .IsFirst([valueIfTrue][,valueIfFalse]) .IsNotFirst([valueIfTrue][,valueIfFalse]) .IsLast([valueIfTrue][,valueIfFalse]) .IsNotLast([valueIfTrue][,valueIfFalse]) .IsPosition(int,[valueIfTrue][,valueIfFalse]) .IsNotPosition(int,[valueIfTrue][,valueIfFalse]) .IsModZero([valueIfTrue][,valueIfFalse]) .IsNotModZero([valueIfTrue][,valueIfFalse]) .IsEven([valueIfTrue][,valueIfFalse]) .IsOdd([valueIfTrue][,valueIfFalse]) .IsEqual(DynamicNode[,valueIfTrue][,valueIfFalse]) .IsDescendant(DynamicNode[,valueIfTrue][,valueIfFalse]) .IsDescendantOrSelf(DynamicNode[,valueIfTrue][,valueIfFalse]) .IsAncestor(DynamicNode[,valueIfTrue][,valueIfFalse]) .IsAncestorOrSelf(DynamicNode[,valueIfTrue][,valueIfFalse])
<b>Dynamic Node Properties</b> .Parent .Id .Template .SortOrder .Name .Visible (requires umbracoNaviHide property) .Url .UrlName .NodeTypeAlias .WriterName .CreatorName .WriterID .CreatorID .Path .CreateDate .UpdateDate .Version .NiceUrl .Level .PropertiesAsList .ChildrenAsList .Position()	<b>Filtering &amp; Ordering &amp; Extensions</b> .Where("Condition",[valueIfTrue,valueIfFalse]) .OrderBy("propertyAlias [desc][,propertyAlias]") .GroupBy("propertyAlias") .Pluck("propertyName") .Take(int) .Skip(int) .Count()	<b>Macro Parameters</b> @Parameter.ParameterName  <b>Media (use for media picker property)</b> .Media("propertyAlias", "mediaPropertyAlias") .Media("propertyAlias")	<b>Permissions</b> .HasAccess() .IsProtected()
<b>DynamicMedia Properties</b> .UmbracoFile .UmbracoSize .UmbracoWidth .UmbrachHeight		<b>Types for Casting and Newing</b> DynamicNode(int nodeId) DynamicMedia(int mediaId) DynamicNodeList dynamic (allows .PropertyAlias notation)	
<b>Custom Property Access (Content &amp; Media)</b> .PropertyAlias ._propertyAlias (recursive) .GetProperty("propertyAlias").Value Notes: casing on property aliases are important. All hyphens must be removed for .Notation When accessing using .Notation you need to ensure you capitalize the first letter unless _recursive	<b>Functions</b> @functions{ public bool isFooBar(string foo, string bar){ return foo == bar; } } @{ var foo = "black"; if(isFooBar(foo,"black")){ <p>Yup</p> } }	<b>Razor Syntax</b> <b>Code Block</b> @{ ... }  <b>Conditionals</b> @if(item.HasValue("bodyText"){ @item.BodyText }else if(item.IsNullOrEmpty("bodyText")){ <p>this item is null</p> }else{ <p>Some other text</p> } }	<b>Comments</b> @* Code comment *@  @switch(condition){ case 1: <p>@item.BodyText</p> break; case 2: <p>@item.Children.First().BodyText</p> break; default: break;
<b>Dictionary</b> @Dictionary .DictionaryItemAlias @Dictionary["dictionaryItemAlias"]		<b>Looping</b> @for(var i = 0; i < 10; i++){ This is record @i }  @foreach(var item in Model.Children){	
<b>@Library Helper</b> <b>Loaders</b> .NodeById(int  string) .MediaById(int  string)  <b>Conditionals</b> .If(booleanProp,valueIfTrue[,valueIfFalse])	<b>Manipulation</b> .Coalesce(value,value[,value ...]) .Concatenate(value,value[,value ...]) .Join(seperator,value,value[,value ...]) .Truncate(htmlString,int[[,bool addEllipsis][,bool treatTagsAsContent]]) .StripHTML(htmlString[, tagsToString])		<b>Visual Studio DynamicNode Intellisense</b> Put the following at the head of the CSHTML file to get some Intellisense support for DynamicNode and DynamicNodeList @using umbraco.MacroEngines @inherits umbraco.MacroEngines.DynamicNodeContext