CTC Smart Client Configurator

Version 1.0.5

Table of Contents

1 Introduction

1.1 What is CTC Smart Client Configurator?

The CTC Smart Client Configurator is the Configurator for the Smart Client Generator from Client Tools Consultancy.

The Configurator is an add-in to the CTC Configurator Framework. It provides the user interface to configure options and features specifically for the Smart Client Generator.

This document should be read in conjunction with the **CTC Configurator Framework** document and the **CTC Smart Client Generator** document.

1.2 The Concept

Typically, a generator creates a fixed, pre-determined user interface application, where users have limited or no influence on what is generated. Customizations have to be applied by modifying the generated user interface application, or by writing a custom generator.

The concept of CTC Generators is to include as many requirements as possible in the generate stage rather than applying modifications to the User Interface after it has been generated.

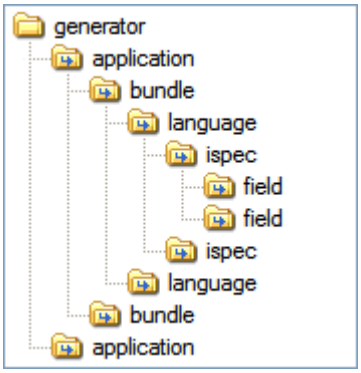
This requires the generator to be very flexible and to provide the means for customizing the result of the generate prior to entering the generate phase.

CTC Generators provide the ability to influence what is generated by configuring features, setting options, customizing Standard Controls, adding Custom Controls and substituting Controls. The generated user interface is still based on the forms and controls being painted in the EAE or AB Suite development environment, however, the CTC Smart Client Configurator allows the developers to specify how each form and control is to be generated at the application, bundle, language, ispec and field level.

Being able to configure and specify the required customization before the generate phase provides a repeatable and automated solution that in most cases will not need further modifications to the generated source.

1.3 Configurable Elements

A configuration consists of configurable elements, and options and features that can be specified for each of these elements. Configurable elements are generator, application, bundle, language, ispec and field, which are specified in a hierarchy as illustrated below.



For information about configurable elements and their hierarchical relationship, refer to the help documentation for the **CTC Configurator Framework**.

The following is a list of items (options/features) that can be configured for the Smart Client Generator within each of the configurable elements (generator, application, bundle, language, ispec and field):

- License Keys
- Custom Controls
- Control Substitutions
- Control Specifications
- Font Substitutions
- Options
- Runtime Configuration

1.4 License Keys

License Keys are required for using the generator and can be specified at the generator level only.

License keys for the Smart Client Generator are specific for the computer on which the generator is installed. A license key is obtained by contacting Client Tools Consultancy.

License keys are stored in this element. When starting the Smart Client Generator for the first time it will request the user to enter a valid license key, evaluation license or full license. Alternatively, the license key can be entered directly into the configuration by adding a License key element to the Smart Client Generator.

When sharing configuration files between work stations by copying a configuration file, the license key element can contain keys that are invalid on the current computer. Invalid license keys are ignored by the generator.

Invalid or expired license keys can be deleted.

1.5 Standard Controls

Standard Controls provide the default look and feel of the controls as they are painted and specified in the EAE and AB Suite Development environments. Standard Controls are built into the Smart Client Generator. They can be used as they are without further customization or they can be customized.

The following is the list of Standard Controls available with the Smart Client Generator:

Control	Description
ButtonGroup	Container for a group of buttons such as Check Box, Push Button and Radio Button that is associated with the same field.
CheckBox	Check Box.
ComboBox	Drop down list box.
Form	Container for controls painted on a form.
Form Page	Outermost container for the form.

Grid	Grid container for CopyFrom regions.
GridHeaderRow	Grid Header Row container for CopyFrom regions.
GridHeaderCell	Grid Header Cell container for CopyFrom regions.
GridRow	Grid Row container for CopyFrom regions.
GridCell	Grid Cell container for CopyFrom regions.
Image	Image.
Label	Label.
Line	Horizontal, vertical and diagonal line.
ListBox	List Box.
Panel	Group container for other controls. AB Suite only.
PushButton	Push Button.
RadioButton	Radio Button.
Rectangle	Rectangular box.
SimpleGrid	SimpleGrid container for CopyFrom regions.
SimpleGridHeader	SimpleGrid Header container for CopyFrom regions.
SimpleGridBody	CimpleGrid Body container for CopyFrom regions.
SimpleGridRow	SimpleGrid Row container for CopyFrom regions.
Teach	Container for controls painted on a teach form.
Teach Page	Outer container for the teach form.
TeachText	Text for the teach form.
TextBox	Text input and Password box.

2 Configuration Items

2.1 Custom Controls

Custom Controls are controls created by either the customer or CTC for a specific purpose. A Custom Control is used when none of the Standard Controls match the requirements.

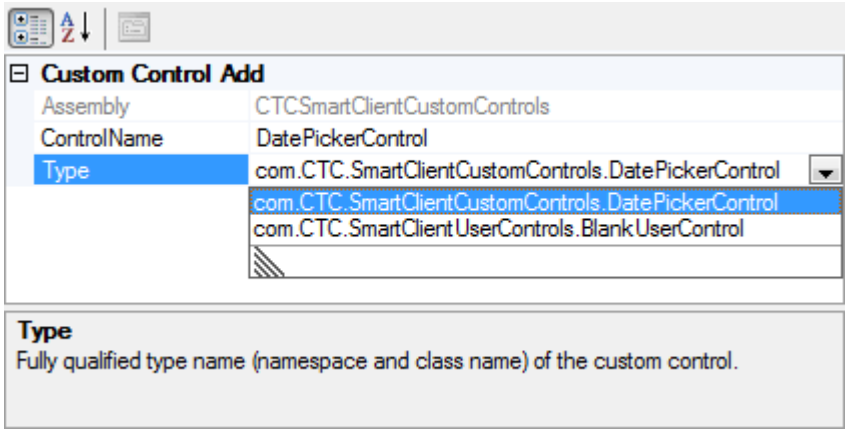
Custom Controls also allow the utilization of third party controls in the generated user interface application. For example, to utilize a Date Picker control from a third party, a custom control can be created as an add-in to the generator, which would generate the correct source specifications and code for the third party control to be executed at runtime in the generated user interface application.

For information on how to create a Custom Control, refer to the **CTC Smart Client Generator** documentation.

A Custom Control is added to the configuration at the generator level only and applies to all configurable items such as application, bundle, ispec and field items within the generator.

The assembly dll containing the Custom Control must be copied to the bin sub-folder of the Component Enabler Root directory, for example, C:\CE3.3\bin.

When adding a Custom Control, the ControlName and Type properties as illustrated below must be specified.



Assembly: This identifies the assembly dll which contains the Custom Control. This is automatically filled when selecting the type.

ControlName: User specified unique name of the Custom Control. When adding a new control type, this is pre-filled with the type name of the control.

Type: This is selected from the dropdown list, which is automatically filled with a list of Custom Controls found within the bin sub-folder of the Component Enabler Root directory. The type specifies the fully qualified type name, which is the namespace and class name of the Custom Control.

2.2 Control Substitutions, Add

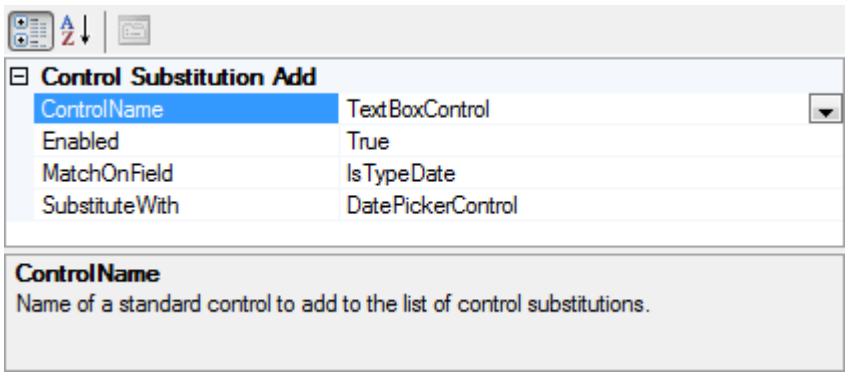
Control Substitutions are used when replacing/substituting Standard Controls with Custom Controls.

Control Substitutions can be added to any level in the hierarchy.

Controls being substituted must be of similar type. For example, when substituting a textbox control with a custom date popup control, the date popup control must be capable of understanding how to generate the control that was originally painted as a textbox.

When control substitutions are inherited from parent levels, control substitutions at the current level are evaluated first. When substituting the same control with different custom controls specifying match conditions, the substitutions will be evaluated in the order they are specified, except for unconditional substitutions which will be evaluated last.

When adding a Control Substitution, the ControlName and SubstituteWith properties as illustrated below must be specified.

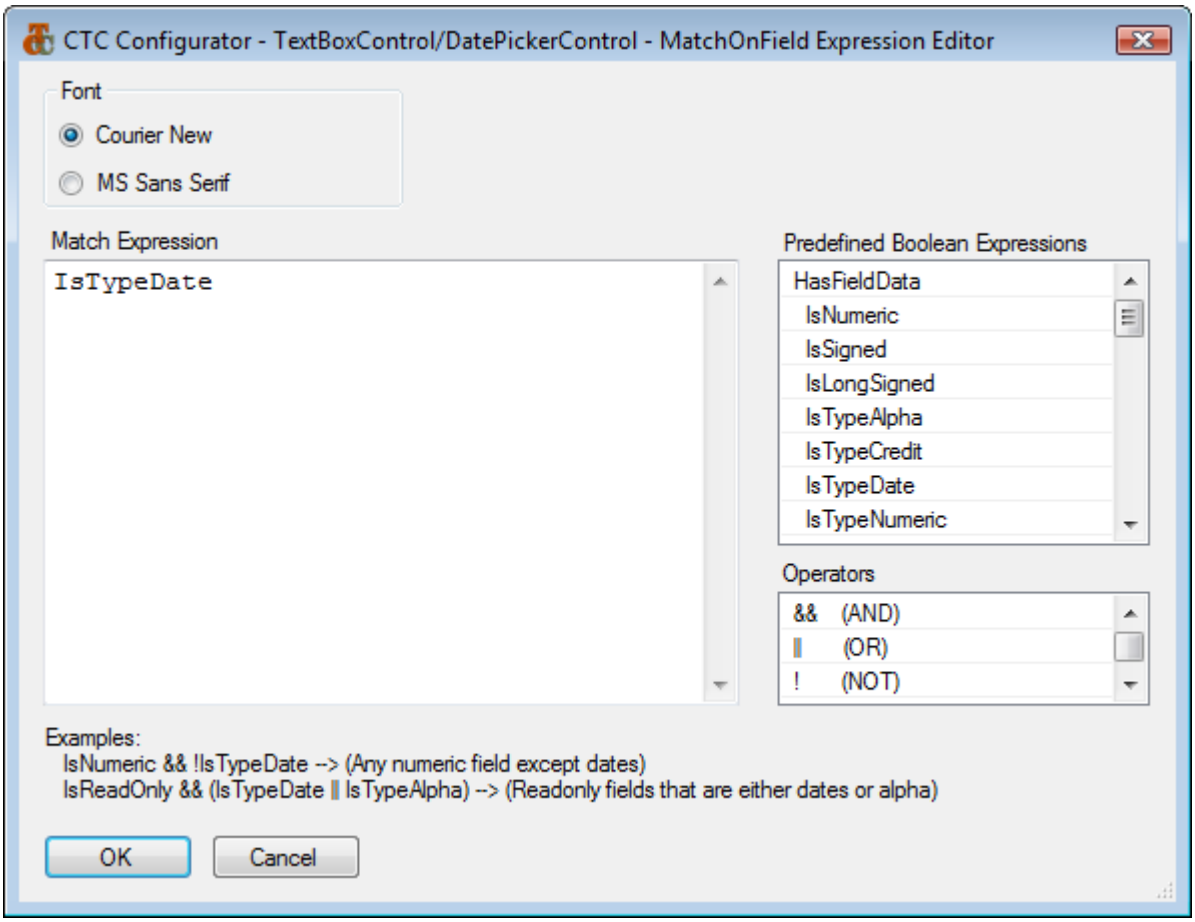


ControlName: This specifies the name of a Standard Control to be substituted. This is selected from the dropdown list, which is automatically filled with a list of available Standard Controls.

Enabled: This property provides a convenient way of removing a control substitution from the generate process without deleting it.

MatchOnField: Optional match expression. MatchOnField is a boolean expression specifying the condition for selecting to which field or fields to apply the substitution. At generate time, the MatchOnField expression is evaluated against the field associated with the particular control and when the expression evaluates true, the substitution is applied. When no expression is specified, control substitution is unconditionally applied.

The expression is specified by selecting the ellipsis button [...]. This will open the Expression Editor window as illustrated below.



An expression is specified using a selection of predefined boolean expressions. Expressions are dragged/dropped or copied/pasted from the list of predefined boolean expressions. Operators such as && (AND), || (OR), ! (NOT), == (Equal), != (Not Equal), > (Greater Than), < (Less Than), >= (Greater Than or Equal) or <= (Less Than or Equal) as well as parentheses ( ) can be used to create complex expressions.

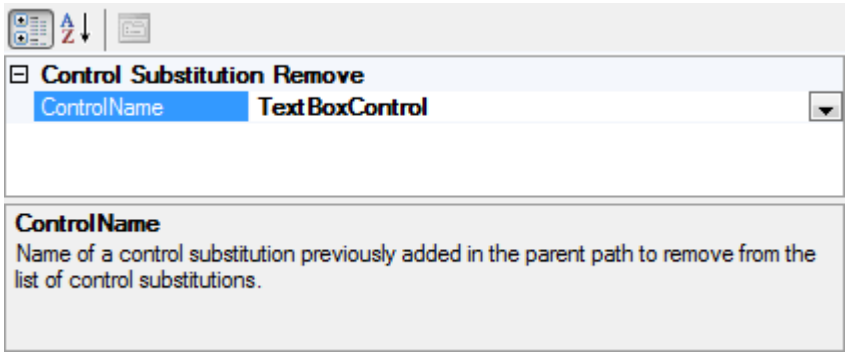
SubstituteWith: This specifies the name of a Custom Control to use instead of the Standard Control. This is selected from the dropdown list, which is automatically filled with a list of Custom Controls currently added to the generator.

2.3 Control Substitutions, Remove

A Control Substitution that has previously been added to a parent in the hierarchy can be removed at any level in the hierarchy by inserting a Control Substitution Remove item.

For example, if a substitution of TextBoxControl with DatePopup has been added at the application level, it may be desirable to generate a specific field as a standard TextBoxControl and not as a DatePopup. In this case, a Control Substitution Remove item can be added to the field specifying to remove the substitution for the TextBoxControl.

The properties window of Control Substitution Remove is illustrated below.



ControlName: This specifies the name of a Standard Control that previously has been substituted at a parent level in the hierarchy. This is selected from the dropdown list, which is automatically filled with a list of available Standard Controls. For convenience, a 'Remove All' item is available in the list. When selecting 'Remove All', all control substitutions inherited from the parent levels will be removed.

2.4 User Controls

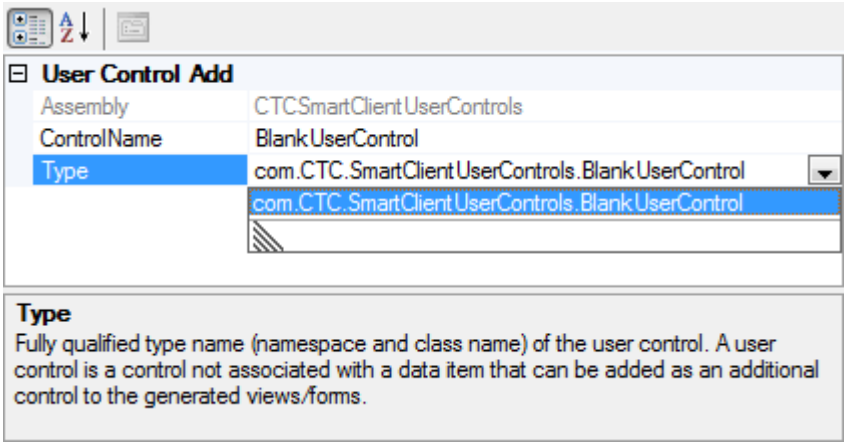
User Controls are controls created by either the customer or CTC for a specific purpose. A User Control is a control that is not associated with a data item in the EAE/AB Suite system. User Controls are used when additional controls are required on the forms to achieve specific tasks.

User Controls are similar to Custom Controls, except User Controls are not associated with data items in the EAE/AB Suite system. For information on how to create User Controls, refer to the 'Creating Own Custom Controls' section in the **CTC Smart Client Generator** documentation.

A User Control is added to the configuration at the generator level only and applies to all configurable items such as application, bundle, ispec and field items within the generator.

The assembly dll containing the User Control must be copied to the bin sub-folder of the Component Enabler Root directory, for example, C:\CE3.3\bin.

When adding a User Control, the ControlName and Type properties as illustrated below must be specified.



Assembly: This identifies the assembly dll which contains the User Control. This is automatically filled when selecting the type.

ControlName: User specified unique name of the User Control. When adding a new control type, this is pre-filled with the type name of the control.

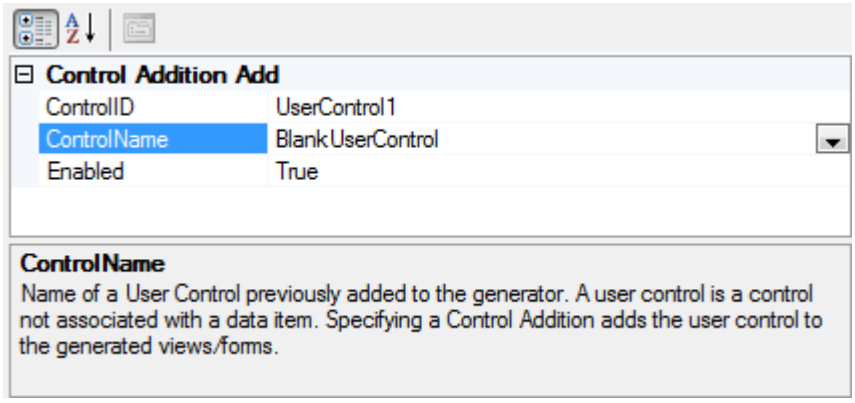
Type: This is selected from the dropdown list, which is automatically filled with a list of User Controls found within the bin sub-folder of the Component Enabler Root directory. The type specifies the fully qualified type name, which is the namespace and class name of the User Control.

2.5 Control Additions, Add

Control Additions are used when adding a User Control to a form.

Control Additions can be added to any level in the hierarchy.

When adding a Control Addition, the ControlID and ControlName properties as illustrated below must be specified.



ControlID: This specifies the ID of the User Control. This is automatically prefilled when a ControlName is selected. Control ID must be unique.

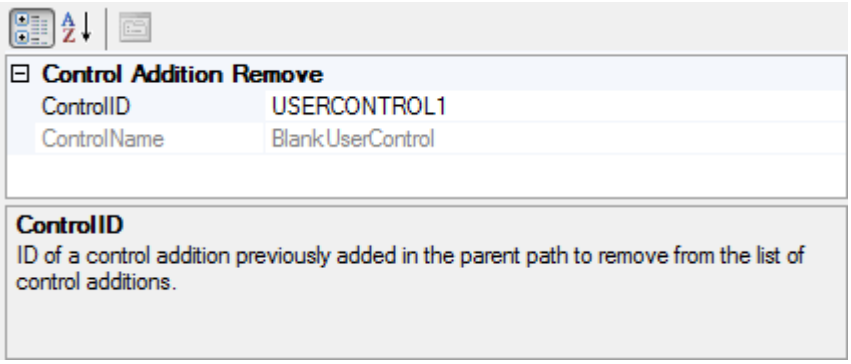
ControlName: This specifies the name of a User Control, previously added to the generator using the User Control Add option, to be added. This is selected from the dropdown list, which is automatically filled with a list of available User Controls.

Enabled: This property provides a convenient way of removing a control addition from the generate process without deleting it.

2.6 Control Additions, Remove

A Control Addition that has previously been added to a parent in the hierarchy can be removed at any level in the hierarchy by inserting a Control Addition Remove item.

The properties window of Control Addition Remove is illustrated below.



ControlID: This specifies the ID of a User Control that previously has been added at a parent level in the hierarchy. This is selected from the dropdown list, which is automatically filled with a list of available User Controls. For convenience, a 'Remove All' item is available in the list. When selecting 'Remove All', all control substitutions inherited from the parent levels will be removed.

ControlName: This specifies the name of the User Control associated with the ControlID. The ControlName property is read-only.

2.7 Control Specifications, Add

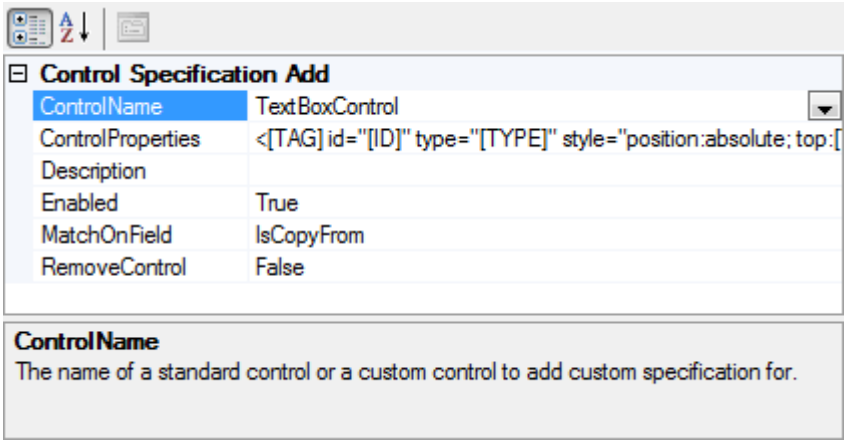
Control Specifications are used to change the default specifications of Standard Controls and Custom Controls. This allows modification of the default look and feel of a control by adding or removing properties and style attributes.

The specification of a control specifies the look and feel of the control. The properties of the control as it is generated are specified here.

Control Specifications can be added to any level in the hierarchy.

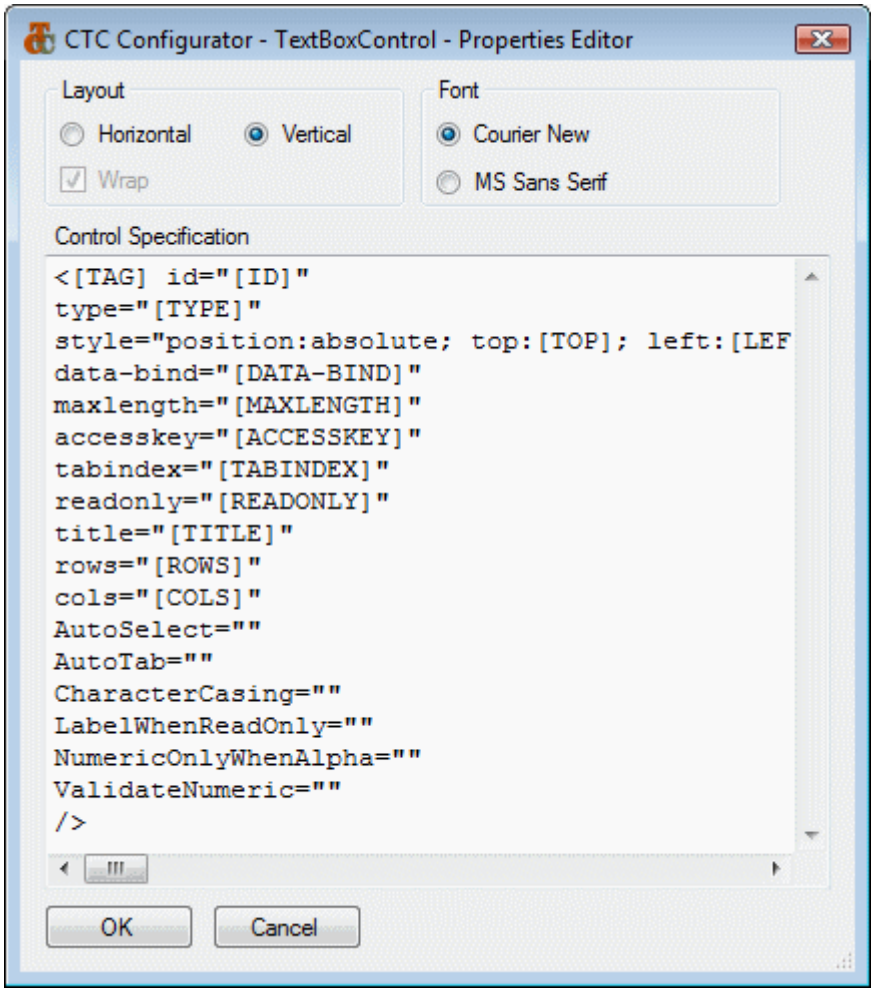
When control specifications are inherited from parent levels, control specifications at the current level are evaluated first. When specifying the same control with different match conditions, the specifications will be evaluated in the order they are specified, except for unconditional specifications which will be evaluated last.

When adding a Control Specification, the ControlName and ControlProperties as illustrated below must be specified.



ControlName: This specifies the name of a Standard Control or a Custom Control for which to specify properties. This is selected from the dropdown list, which is automatically filled with a list of available Standard Controls and Custom Controls.

ControlProperties: This specifies the properties of the control. When selecting a control from the dropdown list of the ControlName property, the default specifications of the control will automatically be shown here. The properties are edited by selecting the ellipsis button which will open the Control Properties Editor window as illustrated below.



The specification of a control defines exactly how the control is generated. Via this editor, the user specifies the properties and their value to be set for the control.

The properties that can be specified on a control are those defined for the HTML controls. Any property available on an HTML control can be added to the specifications of a CTC Standard Control.

The variables specified in square brackets represent the equivalent properties as they are specified for a control when painted in EAE or AB Suite. At generate time, the generator will replace the variables in square brackets with their actual value. If a variable has no value, i.e. it has not been specified in EAE or AB Suite, the whole property will be removed from the control at generate time.

The variables available for a control depend on the control. When initially selecting a control from the ControlName property, the default specifications of the control show the complete set of variables and their usage for the control as seen in the editor window above.

The variable names are case sensitive.

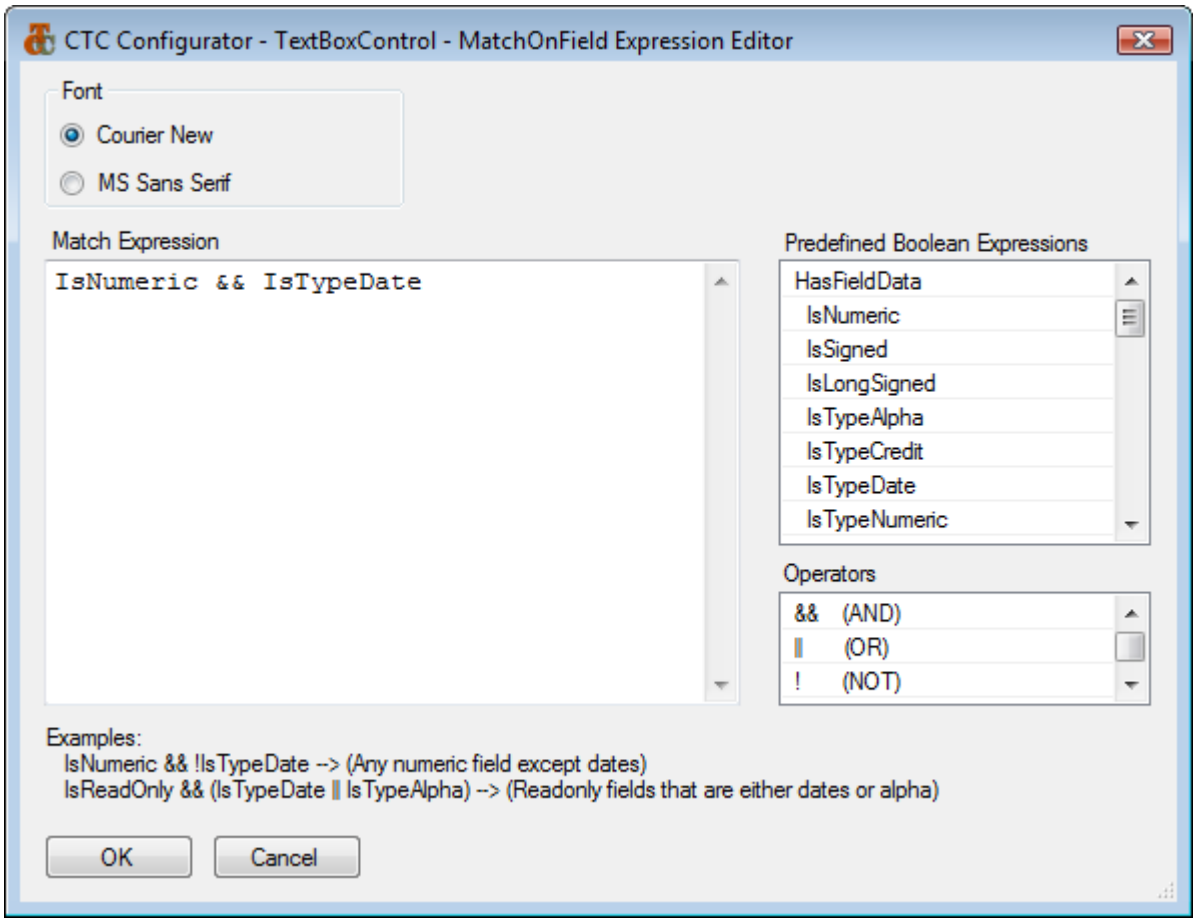
For easy readability, the editor window includes options for the user to choose layout and font.

Description: This property allows adding a description of the control specification to document what has been changed.

Enabled: This property enables/disables the configuration of the specifications for a control. When false, the generator ignores this configuration entry. It provides a convenient way of removing a control specification from the generate process without deleting it.

MatchOnField: Optional match expression. MatchOnField is a boolean expression specifying the condition for selecting to which field or fields to apply the substitution. At generate time, the MatchOnField expression is evaluated against the field associated with the particular control and when the expression evaluates True, the substitution is applied. When no expression is specified, control substitution is unconditionally applied.

The expression is specified by selecting the ellipsis button  which will open the Expression Editor window as illustrated below.



An expression is specified using a selection of predefined boolean expressions. Expressions are dragged/dropped or copied/pasted from the list of Predefined Boolean Expressions. Operators such as && (AND), || (OR), ! (NOT), == (Equal), != (Not Equal), > (Greater Than), < (Less Than), >= (Greater Than or Equal) or <= (Less Than or Equal) as well as parentheses ( ) can be used to create complex expressions.

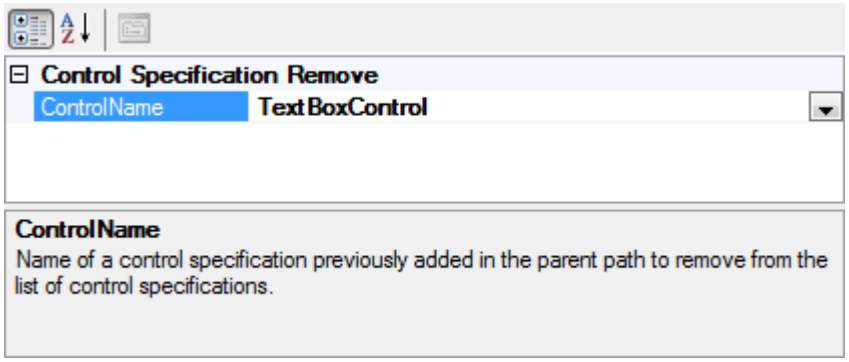
RemoveControl: Specifies whether this control is to be removed from the form. This is a convenient way of excluding a control from being generated.

2.8 Control Specifications, Remove

A Control Specification that has previously been added to a parent in the hierarchy can be removed at any level in the hierarchy by inserting a Control Specification Remove item.

For example, if a specification of a TextBoxControl has been added at the application level, it may be desirable to generate a specific field as a default TextBoxControl. In this case, a Control Specification Remove item can be added to the field to remove the specification for the TextBoxControl.

The properties window of Control Specification Remove is illustrated below.



ControlName: This specifies the name of a Standard Control or Custom Control that previously has been specified at a parent level in the hierarchy. This is selected from the dropdown list, which is automatically filled with a list of available Standard Controls and Custom Controls. For convenience, a 'Remove All' item is available in the list. When selecting 'Remove All', all control specifications inherited from the parent levels will be removed.

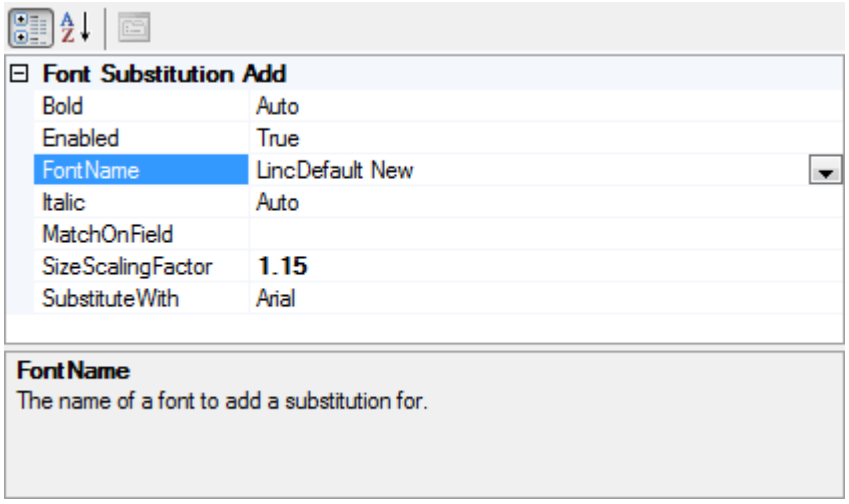
2.9 Font Substitutions, Add

Font Substitutions are used when replacing/substituting fonts that are specified at the time of painting a form, with a different font.

Font Substitutions can be added to any level in the hierarchy.

When font substitutions are inherited from parent levels, font substitutions at the current level are evaluated first. When substituting the same font with different fonts specifying match conditions, the substitutions will be evaluated in the order they are specified, except for unconditional substitutions which will be evaluated last.

When adding a Font Substitution, the properties illustrated below are available.



Bold: This specifies whether to bold the font. The following selections are available:

- Auto – Use the bold setting as specified on the original font.
- On – Bold the font.
- Off – Do not bold the font.

Enabled: This property provides a convenient way of removing a font substitution from the generate process without deleting it.

FontName: This specifies the name of the Font to be substituted. This can be selected from the dropdown list of fonts or typed directly.

Italic: specifies whether to use italic font. The following selections are available:

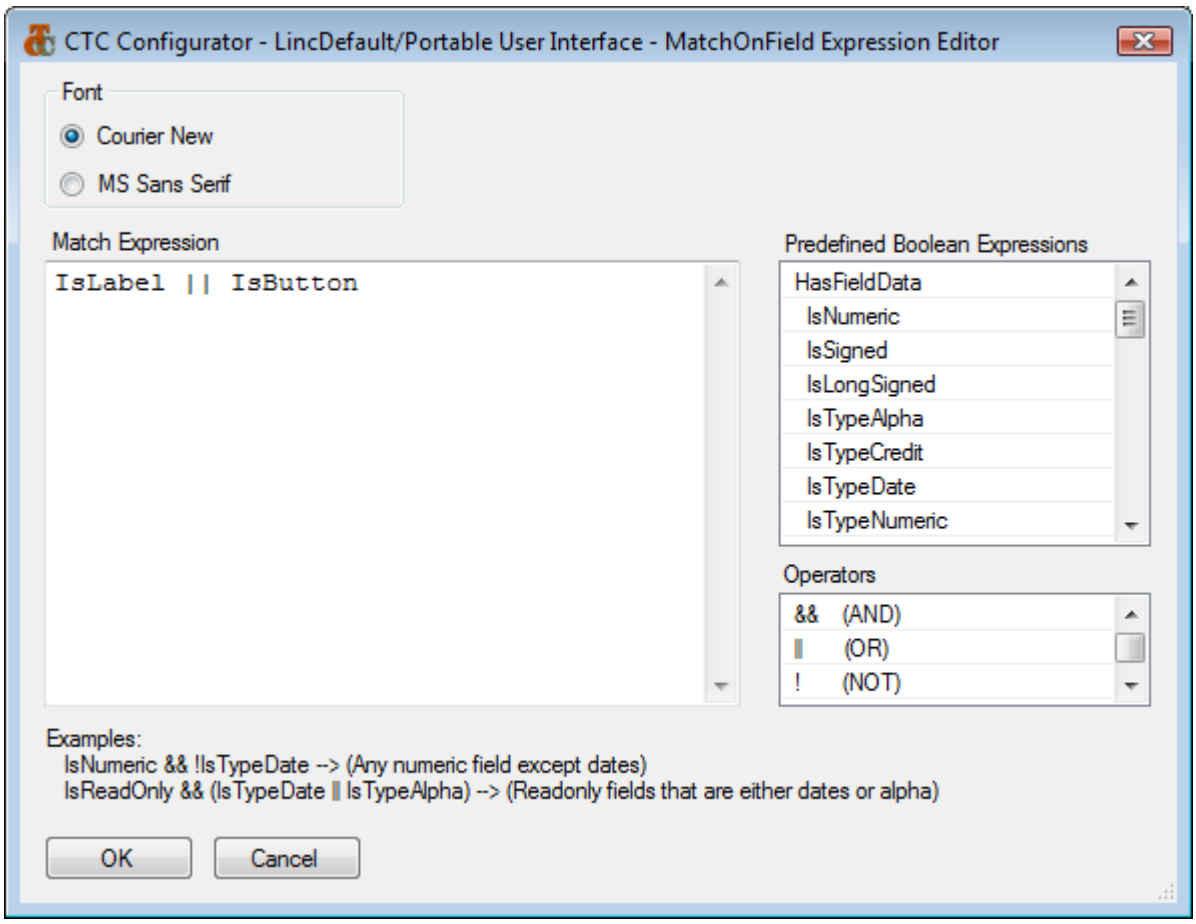
- Auto – Use the italic setting as specified on the original font.
- On – Italic the font.
- Off – Do not italic the font.

SizeScalingFactor: This property specifies the scaling, which is used for increasing/decreasing the font size. When substituting fonts, it may be necessary to increase/decrease the painted font size to get a better match with the substitute font.

SubstituteWith: This specifies the name of a Font to use instead. This can be selected from the dropdown list of fonts.

MatchOnField: Optional match expression. MatchOnField is a boolean expression specifying the condition for selecting to which field or fields to apply the substitution. At generate time, the MatchOnField expression is evaluated against the field associated with the particular font and when the expression evaluates True, the substitution is applied. When no expression is specified, control substitution is unconditionally applied.

The expression is specified by selecting the ellipsis button [...]. This will open the Expression Editor window as illustrated below.



An expression is specified using a selection of predefined boolean expressions. Expressions are dragged/dropped or copied/pasted from the list of Predefined Boolean Expressions. Operators such as && (AND), || (OR), ! (NOT), == (Equal), != (Not Equal), > (Greater

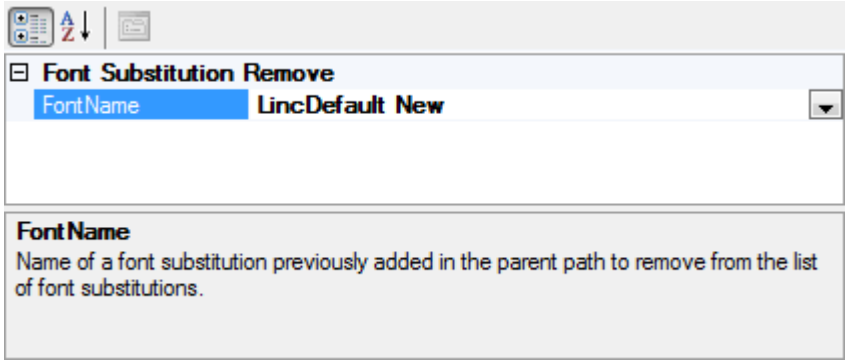
Than), < (Less Than), >= (Greater Than or Equal) or <= (Less Than or Equal) as well as parentheses ( ) can be used to create complex expressions.

2.10 Font Substitutions, Remove

A Font Substitution that has previously been added to a parent in the hierarchy can be removed at any level in the hierarchy by inserting a Font Substitution Remove item.

For example, if a substitution of 'Arial' with 'Courier New' has been added at the application level, it may be desirable to generate a specific field using the painted 'Arial' font and not as 'Courier New'. In this case, a Font Substitution Remove item can be added to the field to remove the substitution for the 'Arial' font.

The properties window of Font Substitution Remove is illustrated below.



FontName: This specifies the name of a font that previously has been substituted at a parent level in the hierarchy. This can be selected from the dropdown list of fonts or typed directly. For convenience, a 'Remove All' item is available in the list. When selecting 'Remove All', all font substitutions inherited from the parent levels will be removed.

2.11 Runtime Configuration

Runtime Configuration specifies parameters to take effect at run time for the generated user interface application.

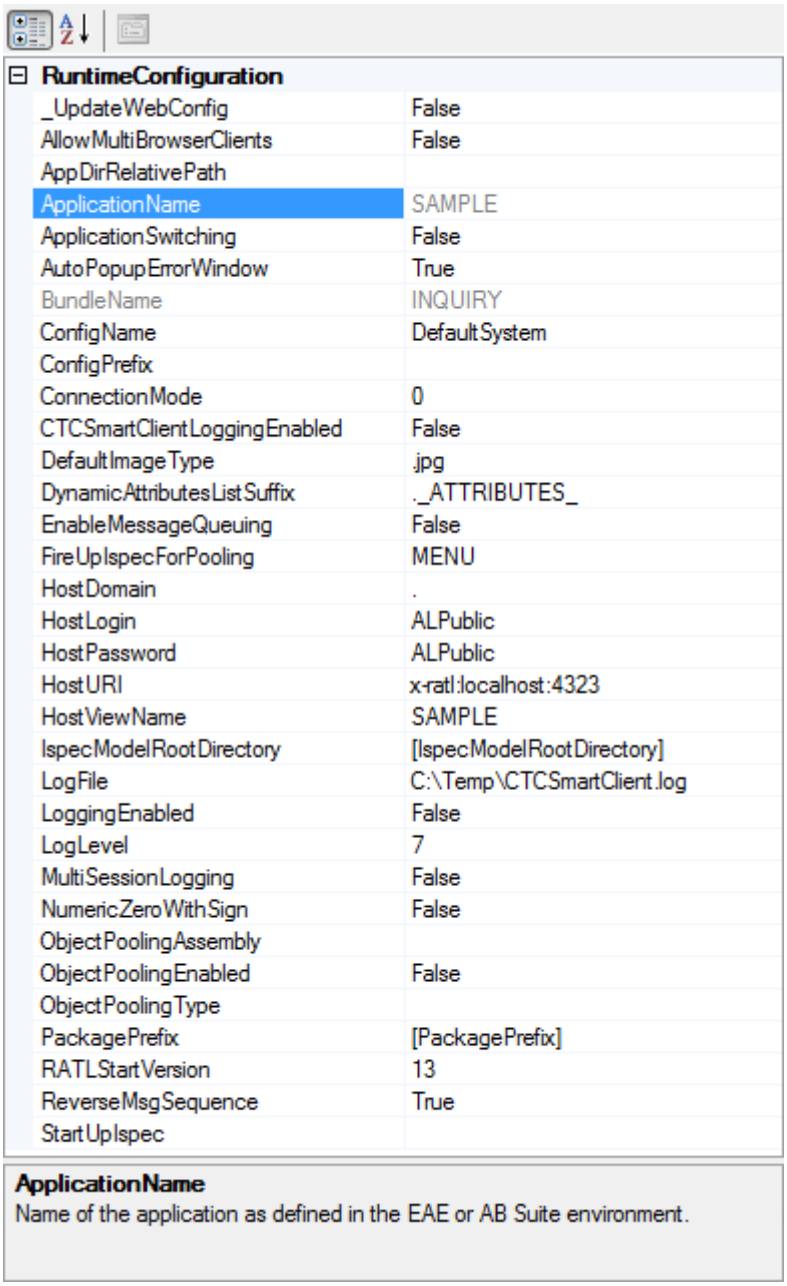
Runtime Configuration can be specified at bundle level only.

In order for the generator to update the Web.config file only when the configuration parameters have changed, the _UpdateWebConfig parameter is automatically set to true when a parameter is changed. However, _UpdateWebConfig can be set false in case it is not convenient to update the Web.config file on the next generate.

At generate time, the Runtime Configuration parameters will be written to the <appSettings> section in the Web.config file when the _UpdateWebConfig option is set. Any existing parameters will be overwritten.

_UpdateWebConfig parameter will automatically be set false by the generator when the Web.config file has been updated. Because the generator updates the configuration file, it is recommended to close the CTC Configurator while running the generator.

A sample Runtime Configuration properties window is illustrated below.



For information on the Runtime Configuration parameters, refer to the relevant section in the documentation for Component Enabler.

