S60

S60 Configuration Tool User Guide

Change history

Version	Date	Status	Comments
2.0	10.03.2009	Approved	
1.0	24.06.2008	Approved	

Contents

1 Introduction to the S60 Configuration Tool

This document is valid from S60 release 3.2.3 onwards.

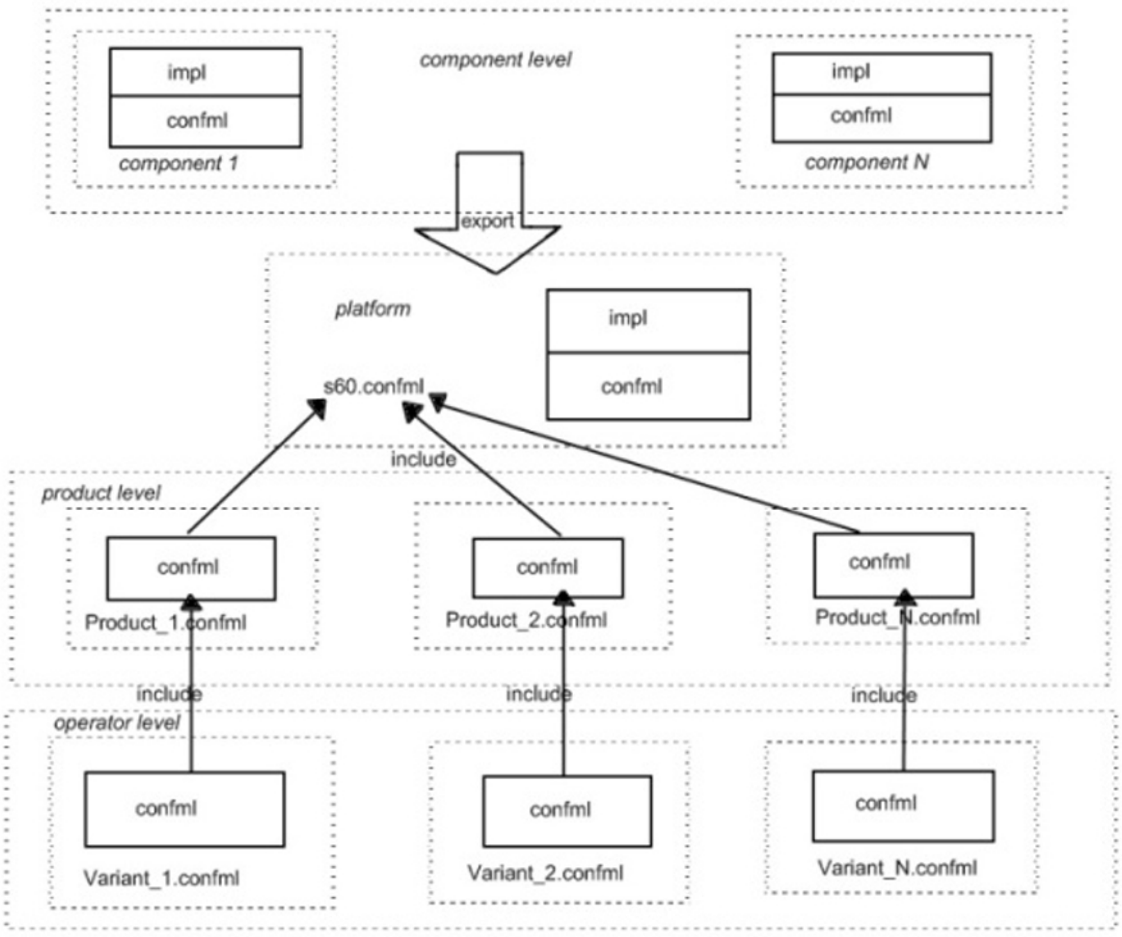
The S60 Configuration Tool allows developers, build engineers, and product and variant engineers to manage component configurations for platform/product integration and variant creation for S60 platform, adaptation, and add-on software. In order for the required configurations to be managed, `ConfML` and `crml` files are used to manage the required information for creating the necessary `Central Repository` keys, etc. This allows:

- definition of various kinds of configurable elements (not only Central Repository keys)
- description of implementations in target environment
- setting/overriding the individual values for elements over their whole lifetime

The S60 Configuration Tool is used in different phases of the platform and device creation process:

- Developers create configurable element definitions and store them with default values.
- Engineers create the necessary files (Central Repository txt files) for a build.
- Product and variant engineers create configurations and change values for individual elements.

The following flowchart illustrates an example of the interaction of these phases.



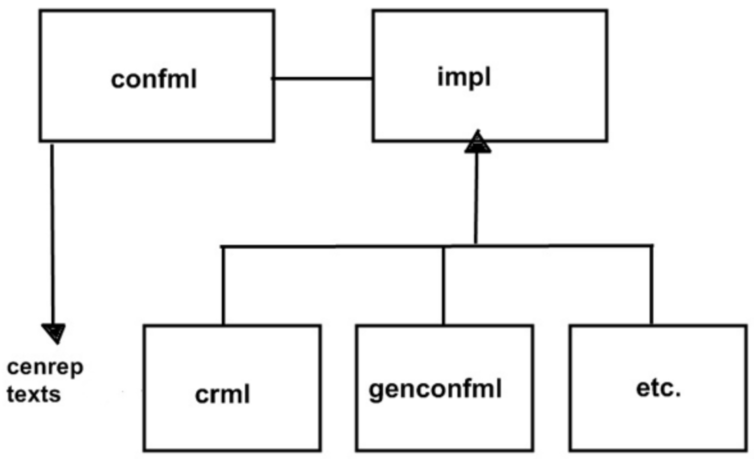
Developers are responsible for documenting the configuration points so that product and variant engineers understand the effect of configuring the element and what the possible values are. Developers own the component's confml and related crml files, and export them to the platform.

The platform gathers all confml files into a single s60.confml master file.

Product programs extend and then modify the platform configurations:

- **Extend**: product programs can add configurations for non-platform components to their product confml file. In addition, the implementation files for the non-platform components are also required.
- **Modify**: change the platform defaults and save the changes in a product confml file.

The following flowchart illustrates the file connections:



- **confml**: Used to manage the configuration of a component, or set of components. These files contain references to underlying crml files through the UidName of the component. If the value for a setting of a component is different than its default value in the crml file, the new value is stored in these files. The platform confml file contains the components released by the platform, as well as the platform settings value if they differ from the defaults. Variant engineers create variant confml files by adding the necessary confml files for any new components, and by assigning new settings values as necessary.
- **impl**: Impl is not a concrete entity, but an abstraction. Crml, genconfml and other (future) files are instances of impl files.
- **cenrep texts**: Cenrep texts are generated by the tool from confml and crml files. Cenrep texts are installed into the device.
- **crml**: Instance of an impl file. Crml files are created by the developers and contain UidName and UidValue, as well as the settings and the value ranges for those settings for the component. These files contain the default settings.
- **genconfml**: Used for settings stored at runtime in arbitrary text format. The file is the root element of the language. Each generated file must be defined in its own Generic Configuration File XML file.

Value of a setting is used in at least one impl (cenrep, genconfml, etc.). The value is associated with impl and the setting itself by reference.

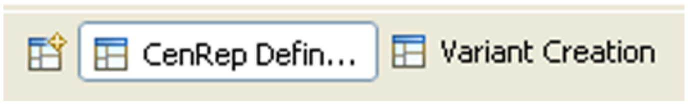
1.1 Navigating in the S60 Configuration Tool

About perspectives

The S60 Configuration Tool has two different perspectives, CenRep Definition perspective and Variant Creation perspective. A perspective defines the initial set and layout of views in the window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources. Perspectives control what is shown in certain menus and toolbars. They define visible action sets, which you can change to customize a perspective.

Selecting a perspective

To open a view in the S60 Configuration Tool, click one of the buttons in the right upper corner of the main window.

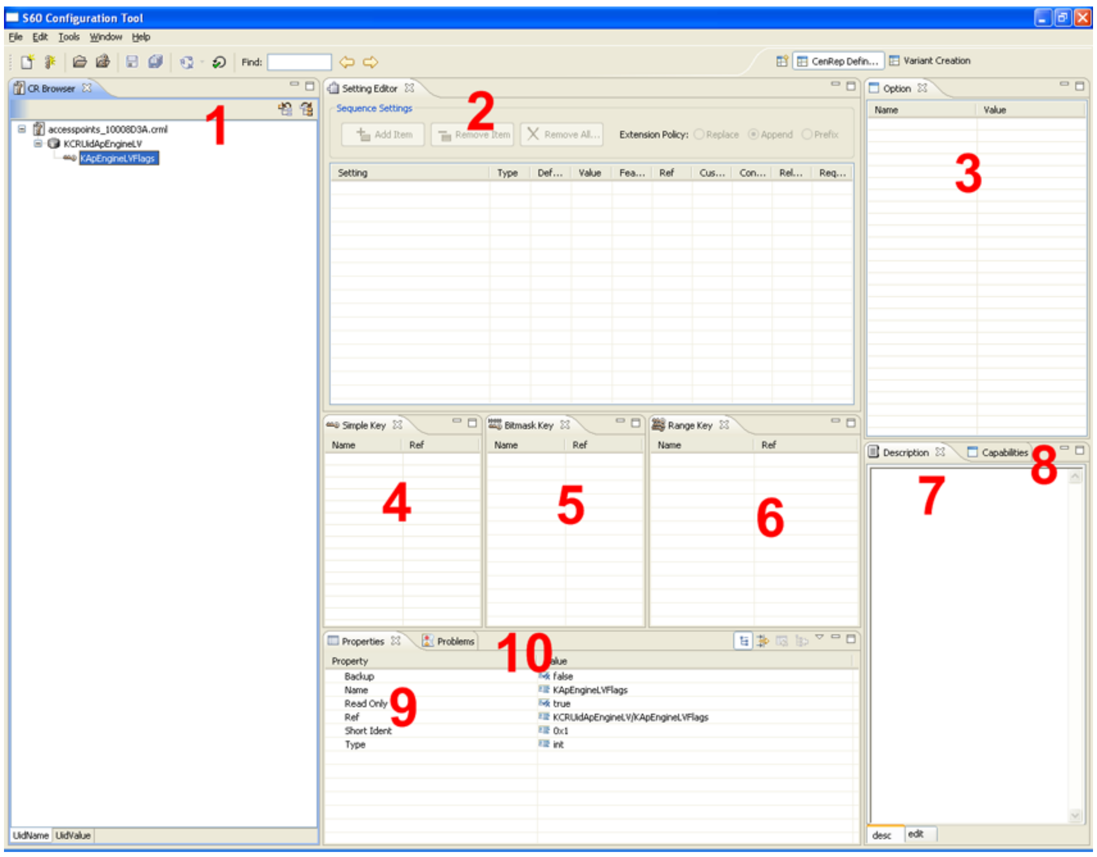


Or

Select the perspective from the menu: **Window --> Open Perspective**. **Open Perspective** -dialog opens from which you can select the correct perspective.

CenRep Definition perspective

The following image illustrates the different parts of the interface in the CenRep Definition perspective:



Property	Value
Backup	false
Name	KApEngineLVFlags
Read Only	true
Ref	KCRUidApEngineLV/KApEngineLVFlags
Short Ident	0x1
Type	int

1 The CR Browser -view is for managing CenRep Definition projects.

2 The Setting Editor -view is for modifying the configuration's setting parameters.

3 The Option -view contains the possible values for the setting.

4 The Simple Key -view shows all simple keys of the selected repository.

5 The Bitmask Key -view shows all bitmask keys of the selected repository.

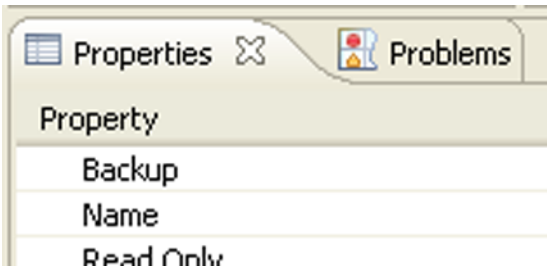
6 The Range Key -view shows all range keys of the selected repository.

7 The Description -view contains an overview of what the selected element is and what it is used for.

8 The Capabilities -view is for setting the capabilities of a repository or a key.

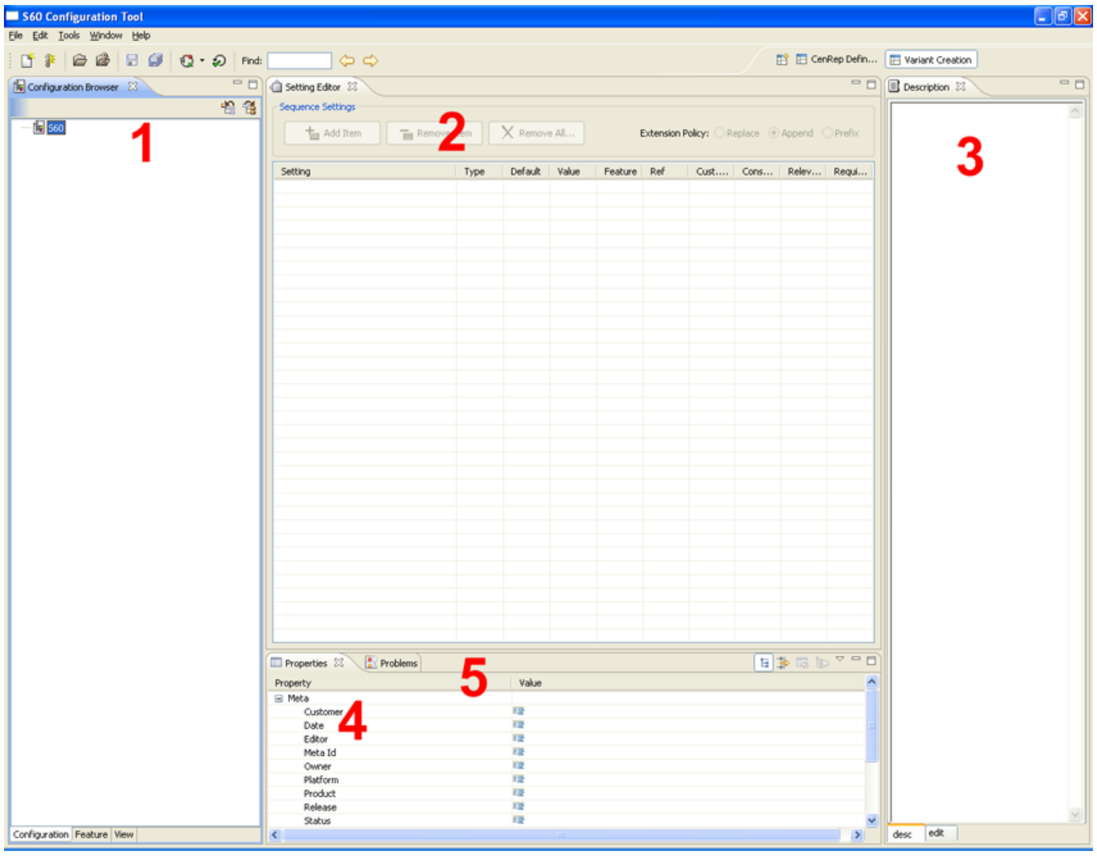
9 The Properties -view is used for modifying the additional values for the selected element.

10 The Problems -view shows the error and warning messages.



Variant Creation perspective

The following image illustrates the different parts of the interface in the Variant Creation perspective:



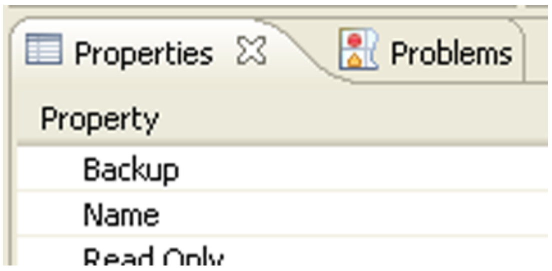
1 The Configuration Browser -view is for managing Variant Creation projects.

2 The Setting Editor -view is for modifying the configuration's setting parameters.

3 The Description -view contains an overview of what the selected element is and what it is used for.

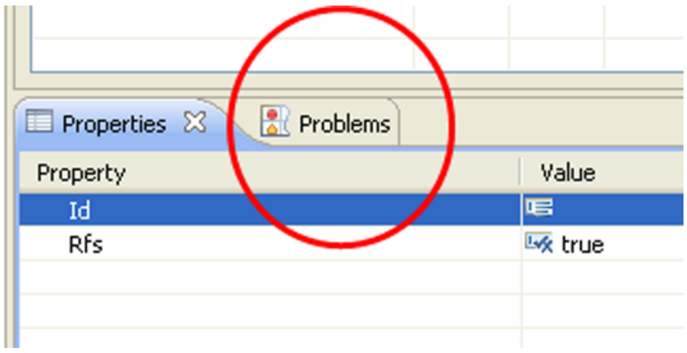
4 The Properties -view is used for modifying the additional values for the selected element.

5 The Problems -view shows the error and warning messages.



If you close a view in the S60 Configuration Tool, select **Window** --> **Show View** --> <desired view> to display it again. For a list of available key shortcuts, select **Help** --> **Key Assist**.

Note that some of the views are stacked or hidden by another view. To bring a view forward, click the appropriate tab:



Property	Value
Id	
Rfs	true

2 Installing the S60 Configuration Tool

The S60 Configuration Tool is a standalone application that has been built on the Eclipse framework. The application uses Java Run-time Environment 1.6 which you need to install separately.

There are two possible versions of the S60 Configuration Tool:

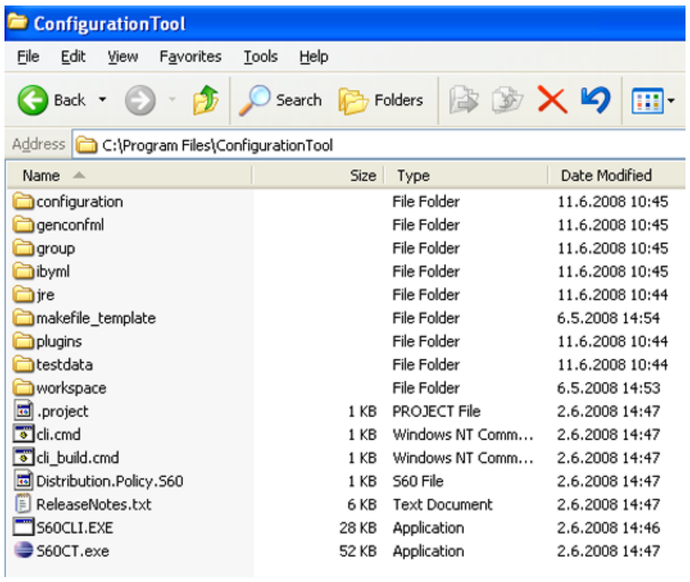
- The S60 Configuration Tool with Graphical User Interface (GUI) provides full functionality for creating and modifying configurations and generating central repositories (cenreps).
- The S60 Configuration Tool with Command Line Interface (CLI) provides a subset of CT functionality for generating cenreps from ready configurations.

Installing and starting the S60 Configuration Tool

1. To install the S60 Configuration Tool, you must download it and place it into a directory where you want to use the it.
2. To start the tool, run "S60CT.exe".

Contents of the package

A properly installed S60 Configuration Tool has the following contents:



C:\Program Files\ConfigurationTool

Name	Size	Type	Date Modified
configuration		File Folder	11.6.2008 10:45
genconfml		File Folder	11.6.2008 10:45
group		File Folder	11.6.2008 10:45
ibyml		File Folder	11.6.2008 10:45
jre		File Folder	11.6.2008 10:44
makefile_template		File Folder	6.5.2008 14:54
plugins		File Folder	11.6.2008 10:44
testdata		File Folder	11.6.2008 10:44
workspace		File Folder	6.5.2008 14:53
.project	1 KB	PROJECT File	2.6.2008 14:47
cli.cmd	1 KB	Windows NT Comm...	2.6.2008 14:47
cli_build.cmd	1 KB	Windows NT Comm...	2.6.2008 14:47
Distribution.Policy.S60	1 KB	S60 File	2.6.2008 14:47
ReleaseNotes.txt	6 KB	Text Document	2.6.2008 14:47
S60CLI.EXE	28 KB	Application	2.6.2008 14:46
S60CT.exe	52 KB	Application	2.6.2008 14:47

The S60 Configuration Tool root folder contains:

- (.project)
- The command line intrepretator for the application CLI executable (cli.cmd)
- The command line interpretator for the application CLI (for build) executable (cli_build.cmd)
- (Distribution.Policy.S60)
- Release notes (Release.Notes.txt)
- The CLI version executable (S60CLI.EXE)

  Note: This should not be executed.
- The application GUI executable (S60CT.exe)
- The root folder also contains the following directories:

  configuration

- The directory for S60 Configuration Tool settings.

genconfml

- A directory for XSLT transformation definitions used for generating configurations.

group

ibyml

- A directory for ibyml definitions that are used when the tool generates files.

makefile_template

- A directory for META and MK files.

plugins

- A directory for Java binaries used by the tool.

testdata

- A directory for testdata for the S60 Configuration Tool.

workspace

- A directory for tool project settings.

3 Creating variants

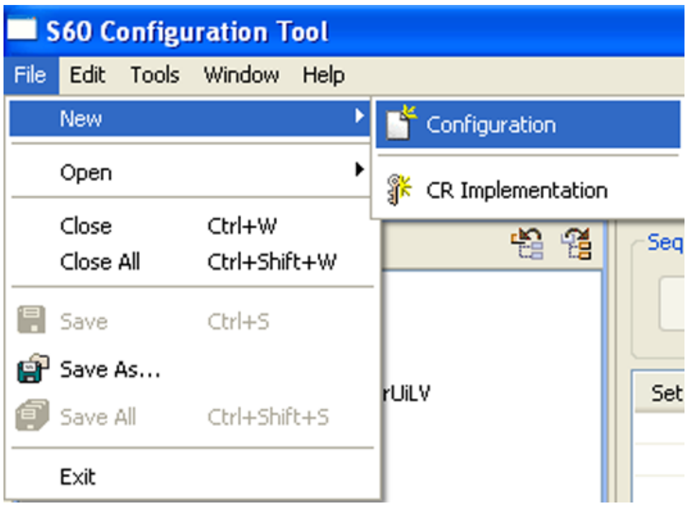
3.1 Creating a new configuration based on an existing configuration

Overview

- Step 1: Create a configuration for your product.
- Step 2: Change the settings for the default master confml file and add new components as necessary.

3.1.1 Creating a new configuration file

1 To create a new configuration file, select **File** --> **New** --> **Configuration**.



A **New Configuration** dialog opens.

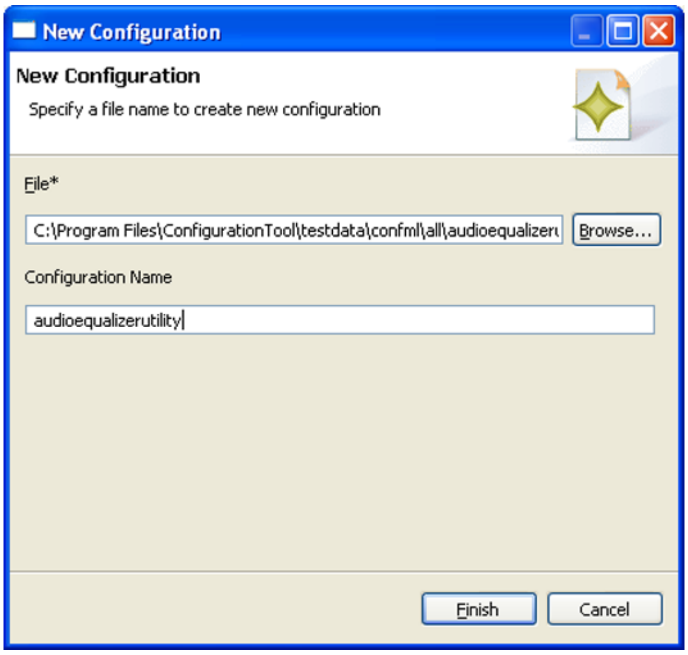
2 Enter the file path and name in the **File** text field. The file extension must be .confml.

Or

Select **Browse...** to open the file dialog and navigate to the desired folder. Type the name of the new configuration in the **File Name** text field and select **Open**.

Note: The file name should only contain lowercase letters and no spaces.

3 In the **New Configuration** dialog, enter a name for the configuration.

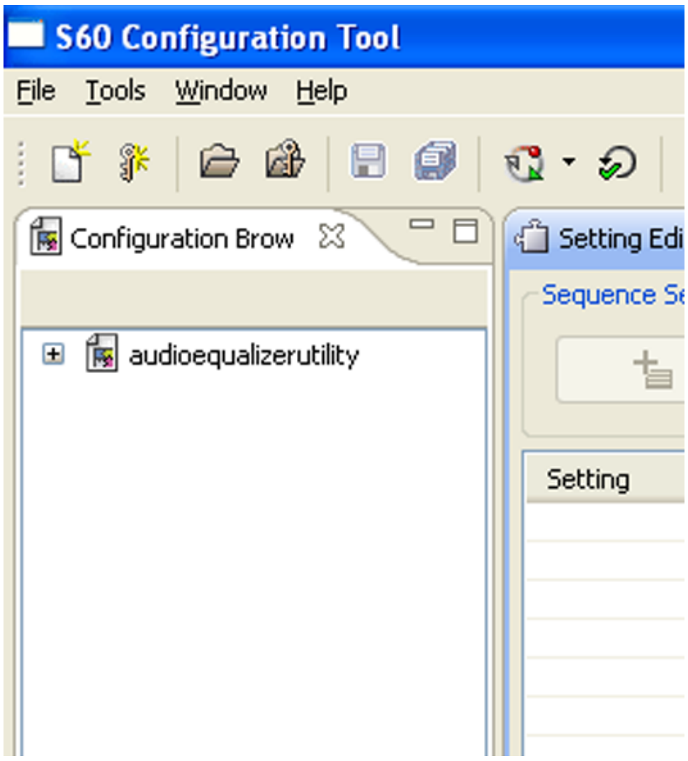


C:\Program Files\ConfigurationTool\testdata\confml\all\audioequalizeru

audioequalizerutility

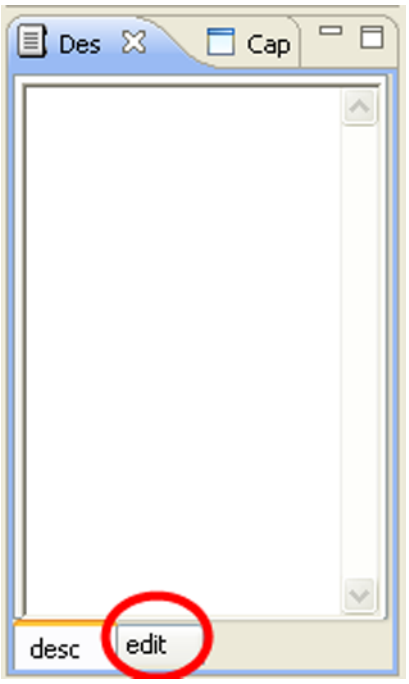
4 When both fields are filled, select **Finish**.

A new configuration is created and visible in the **Configuration Browser** view.



audioequalizerutility

5 A configuration description can be added by using the **Edit** tab in the bottom of the **Description** view.



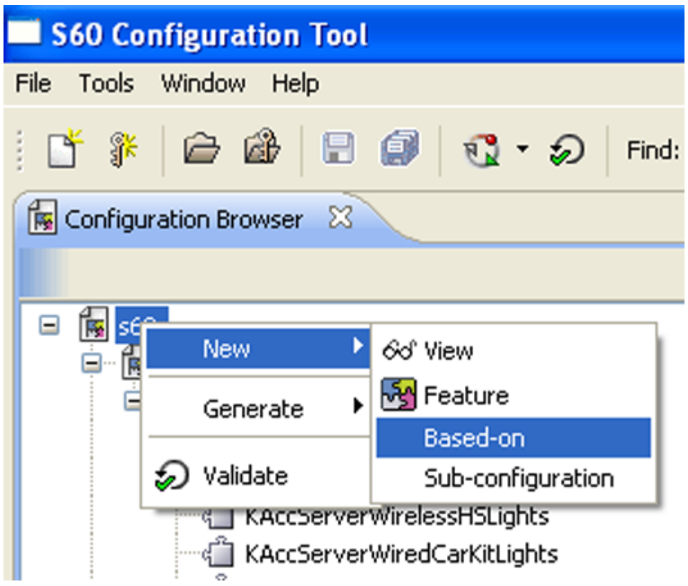
3.1.2 Adding a new based-on configuration

You must have an existing configuration file in order to create a based-on configuration.

1 To create a based-on configuration, select an existing configuration from the **Configuration Browser** view.

2 Right-click on the new configuration to open the popup menu.

3 Select **New** --> **Based-on**.

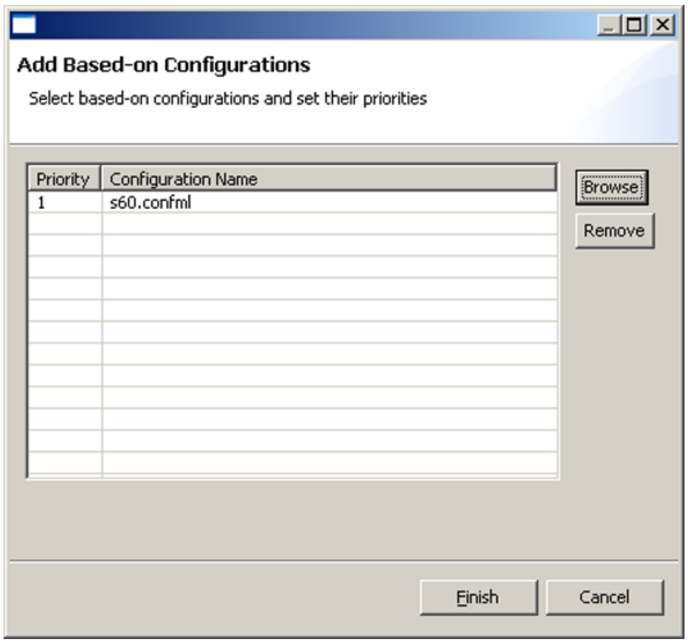


Add Based-on Configurations dialog opens.

4 Select **Browse...** to open the **File** dialog.

5 Find and select the confml file that your new configuration will be based on. Select **Open**.

A based-on configuration is added to the **Add Based-on Configurations** dialog.


Add Based-on Configurations

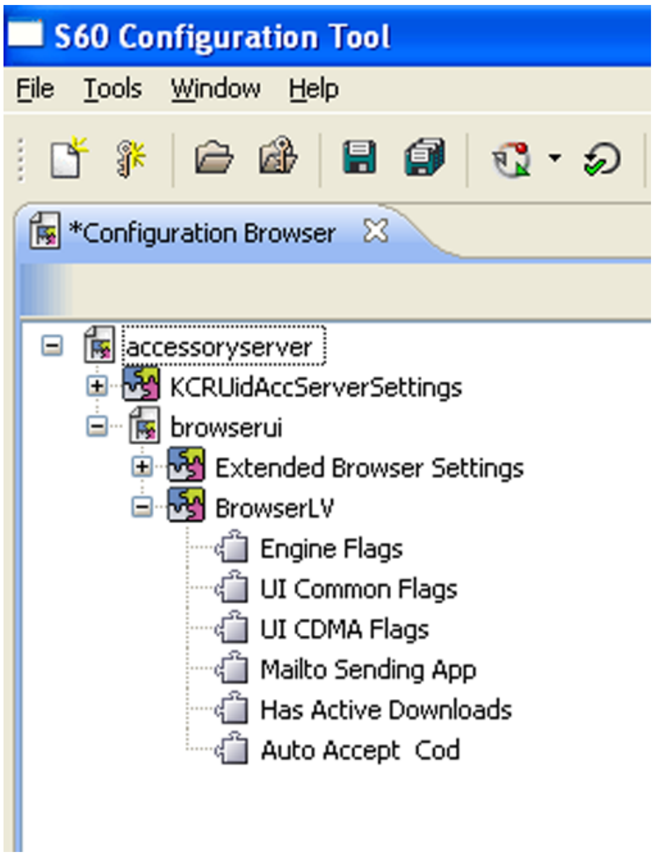
Select based-on configurations and set their priorities

Priority	Configuration Name
1	s60.confml

Note: You can add other based-on configurations in the same way.

6 Select **Finish** to add the based-on configuration(s).

The based-on configurations are added to the Configuration Browser. They are visible under your new configuration.



3.1.3 Importing existing variant data

The S60 Configuration Tool allows variant engineers to import VariantData.xml as it is created by the Customization Tool. Values from VariantData.xml file are copied into the CONFML file only if corresponding setting can be found in the configuration hierarchy. The S60 Configuration Tool can split automatically single value from the VariantData.xml when the values represents a bitmask. The user can choose to import a single VariantData.xml or import the whole chain of VariantData.xml.

Import Single VariantData

Single import takes values only from the selected VariantData.xml. Values defined in the included configuration are ignored.

Import VariantData chain

Values are taken from the whole VariantData chain and applied to a single configuration. Values are taken according to the priority rule defined for VariantData.xml.

GUI version

Use pop-up menu of the current root configuration.

CLI version

cli.cmd -master <file> [-vds|-vdc [-save] <path to VariantData.xml>]

-vds <path to VariantData.xml> import single VariantData. Values are applied but not saved.

-vdc <path to VariantData.xml> import chain of VariantDatas. Values are applied but not saved.

-save applies and saves VariantData values into the master configuration.

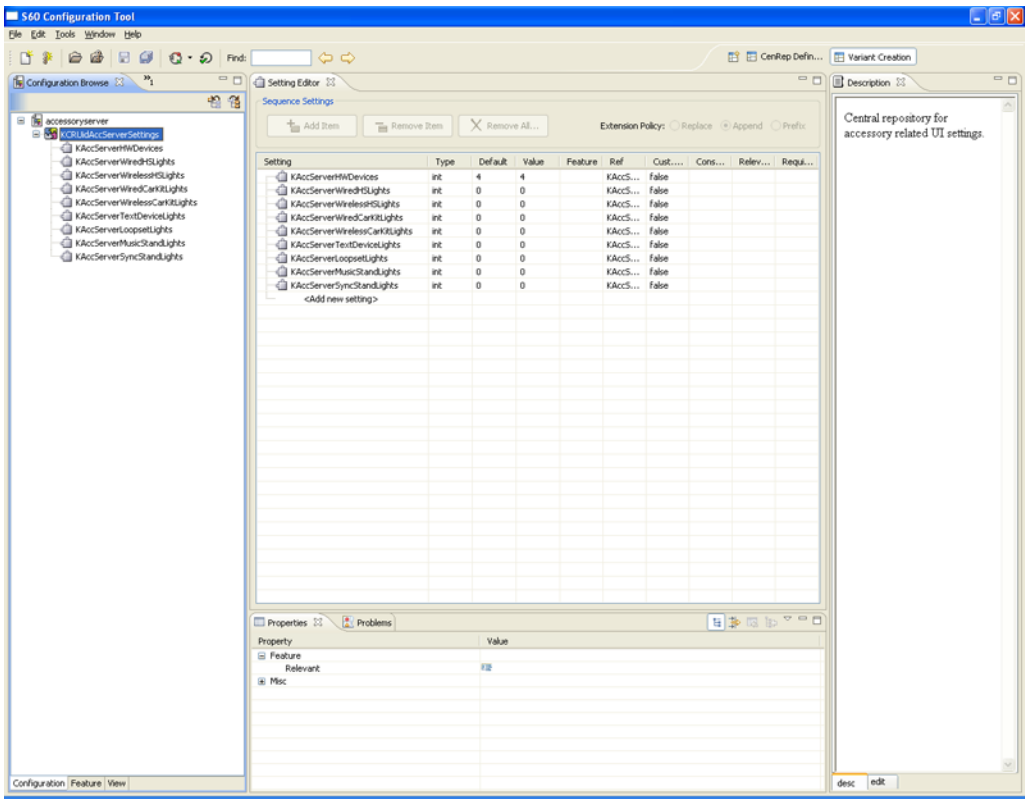
3.1.4 Changing the setting values

1 To change the setting values, select the setting from the Configuration Browser.

2 Select the feature that contains the setting.

All the settings under the selected feature are visible in the Setting Editor.

3 Double-click the **Value** cell of a certain setting and enter a new value.



Setting	Type	Default	Value	Feature	Ref	Cust....	Cons...	Relev...	Requi...
KAccServerHWDevices	int	4	4		KAccS...	false			
KAccServerWiredHSLights	int	0	0		KAccS...	false			
KAccServerWirelessHSLights	int	0	0		KAccS...	false			
KAccServerWiredCarKitLights	int	0	0		KAccS...	false			
KAccServerWirelessCarKitLights	int	0	0		KAccS...	false			
KAccServerTextDeviceLights	int	0	0		KAccS...	false			
KAccServerLoopsetLights	int	0	0		KAccS...	false			
KAccServerMusicStandLights	int	0	0		KAccS...	false			
KAccServerSyncStandLights	int	0	0		KAccS...	false			
<Add new setting>									

Property	Value
Feature	
Relevant	
Misc	

Central repository for accessory related UI settings.

Modify all the necessary settings in the same manner. Notice that the way of editing a value depends on the setting type.

3.1.5 Saving the changes

1 To save the changes you have made, select **File --> Save** (or use shortcut Ctrl+S).

All the changes are saved to the new configuration. For example, the <product>.confml looks like this:

```
<?xml version="1.0" encoding="UTF-16"?> http://www.s60.com/xml/confml/1>
        <data>
                <CommonMessagingUI>
                        <NumberOfMsgInSentItem>50NumberOfMsgInSentItem>50>
                </CommonMessagingUI>
        </data> s60.confml#/"/>
</configuration>
```

4 Developing configurations

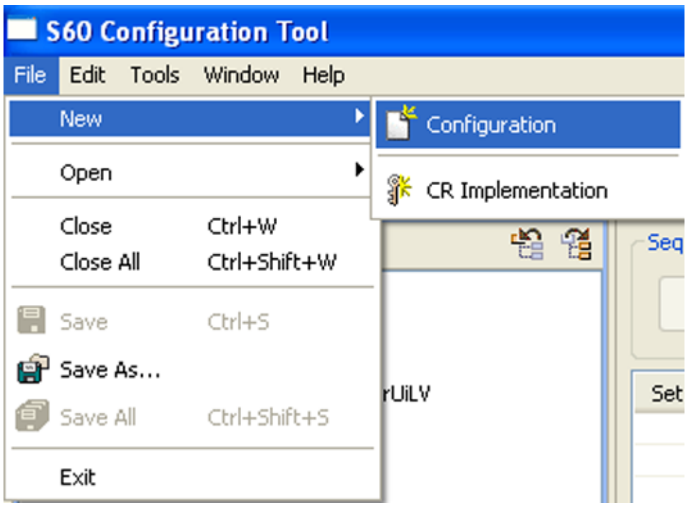
The developers are responsible for creating confml files for their components.

4.1 Creating a new configuration, feature and setting

This is the procedure used when you are not creating a device based on a platform delivery.

Creating a new configuration file

1 To create a new configuration file, select **File --> New --> Configuration**.



A **New Configuration** dialog opens.

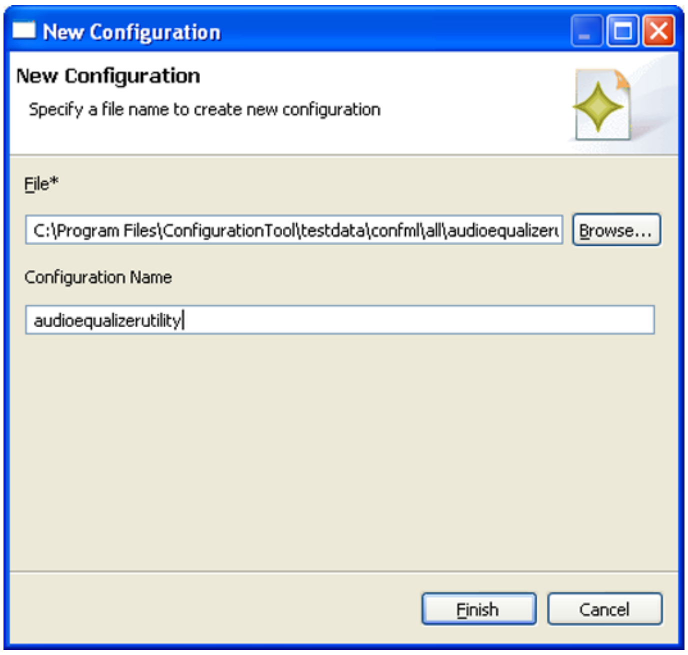
2 Enter the file path and name in the **File** text field. The file extension must be .confml.

Or

Select **Browse...** to open the file dialog and navigate to the desired folder. Type the name of the new configuration in the **File Name** text field and select **Open**.

Note: The file name should only contain lowercase letters and no spaces.

3 In the **New Configuration** dialog, enter a name for the configuration.


New Configuration

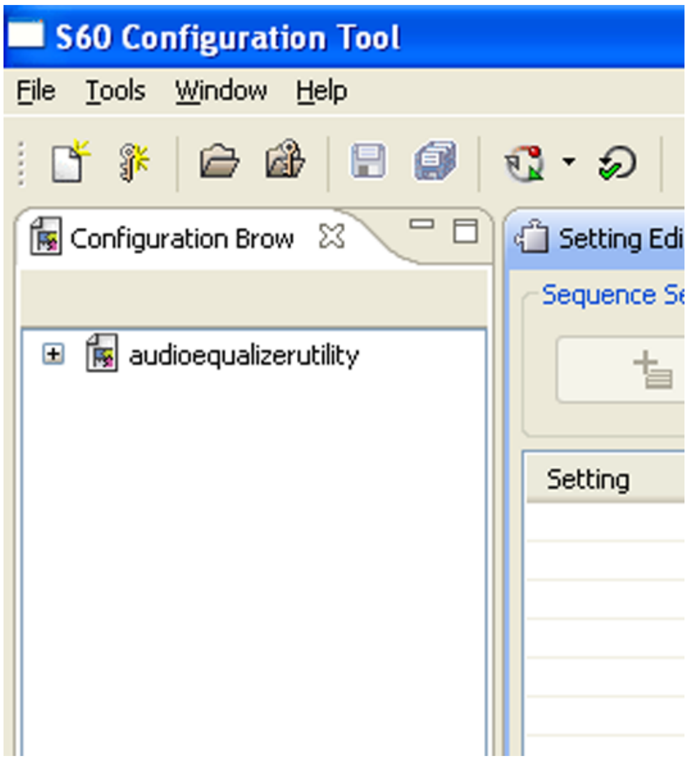
Specify a file name to create new configuration

File*: C:\Program Files\ConfigurationTool\testdata\confml\all\audioequalizerı

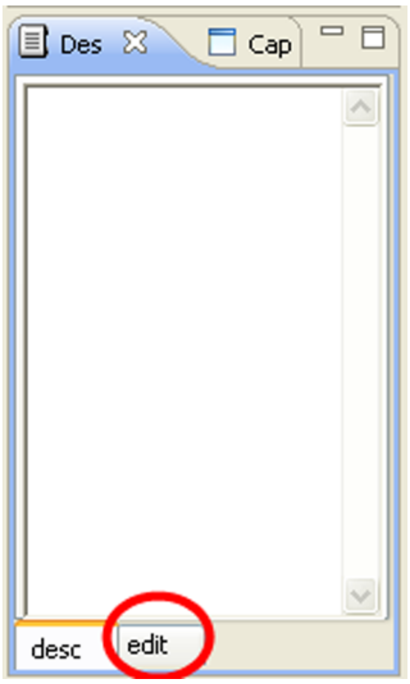
Configuration Name: audioequalizerutility

4 When both fields are filled, select **Finish**.

A new configuration is created and visible in the **Configuration Browser** view.



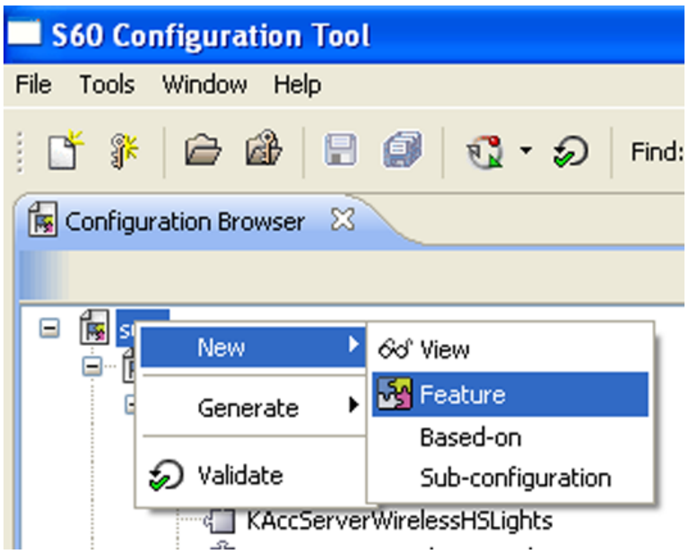
5 A configuration description can be added by using the **Edit** tab in the bottom of the **Description** view.



Adding a new feature

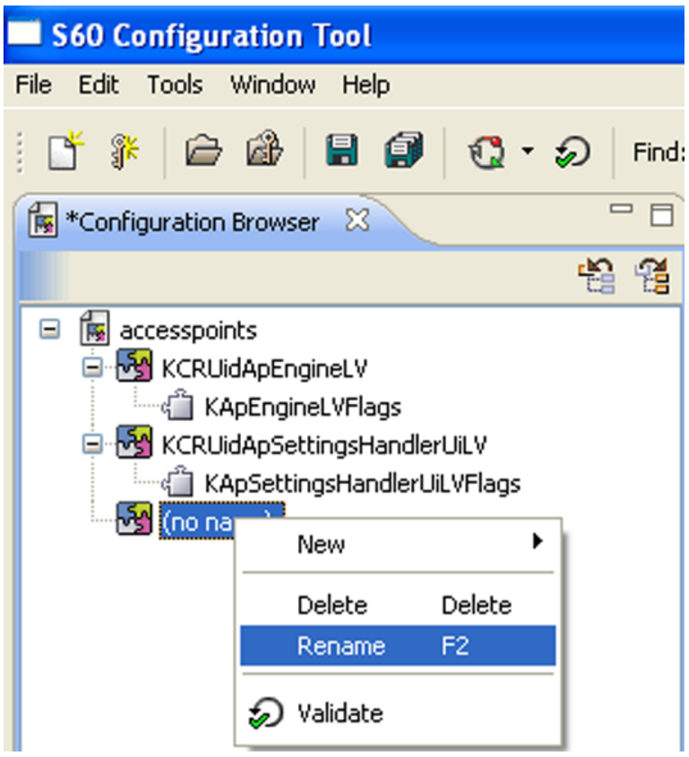
1 To add a new feature, right-click on a configuration (in the **Configuration Browser** view).

2 Select **New --> Feature**.



A new (no name) feature is created under the configuration.

3 Right-click on the created feature and select **Rename** to give a new name for the feature.

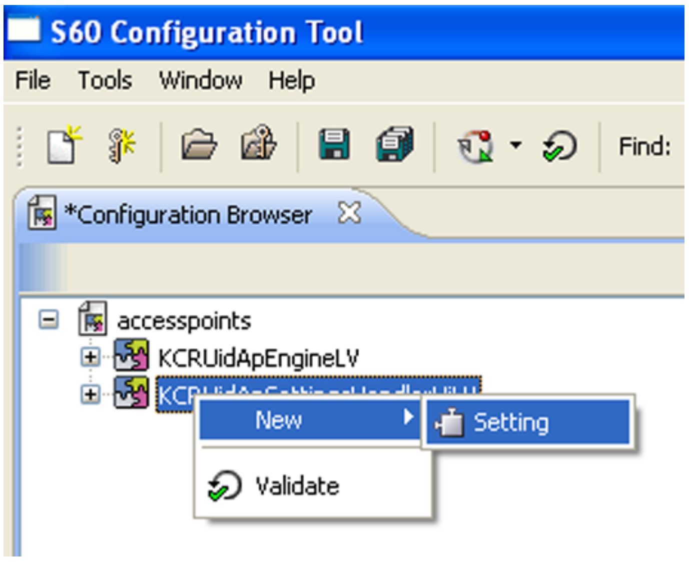


4 A feature description can be added by using the **Edit** tab in the bottom of the **Description** view.

Adding a new setting

1 To add a new setting, right-click on a feature (in the **Configuration Browser** view).

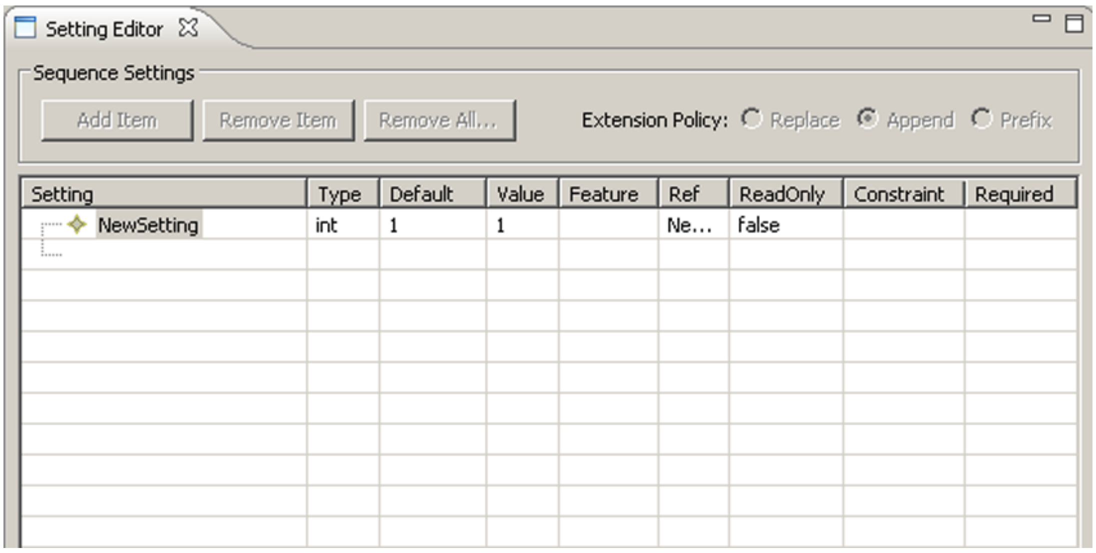
2 Select **New --> Setting**.



3 You can rename the setting by selecting a feature and double-clicking on the setting in the Setting Editor or by using the right-click menu on the setting in the Configuration Browser. When you rename a setting, the reference will be the same, but without the spaces. A setting description can be added by using the **Edit** tab in the bottom of the **Description** view.

Setting parameters

1 To change the setting parameters, select a feature in the Configuration Browser and then double-click an element in the Setting Editor.



Setting	Type	Default	Value	Feature	Ref	ReadOnly	Constraint	Required
NewSetting	int	1	1		Ne...	false		

Note: Variant engineers cannot do all the same modifications as build engineers, for example change the name of the setting.

Setting parameters are explained in the following:

1 Type
- folder - Value is directory in the file system.
- file - Value is file in the file system.
- sequence - Value is sequence of settings. Settings in the sequence can be edited by expanding the setting template.
- int - Value is integer.
- string - Value is string.
- boolean - Value is true or false.
- selection - Value is one item from the predefined list of items. Items in the selection can be edited by expanding the setting template.
- real - Value is floating point.

2 Default - Default value for the setting. If the setting does not have a value then this is used instead.

3 Value - Value for the setting.

4 Feature

5 Ref - Reference for the CRML setting. It is a unique identifier for the feature.

6 Customer Configurable - The customer can change the value if it is true.

7 Constraint

8 Relevant

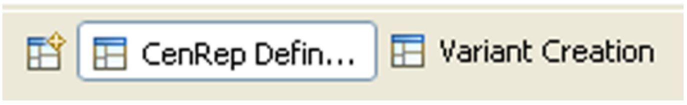
9 Required

Note: At least one of these parameters must be selected if you want to define a setting.

4.2 Creating a new central repository implementation

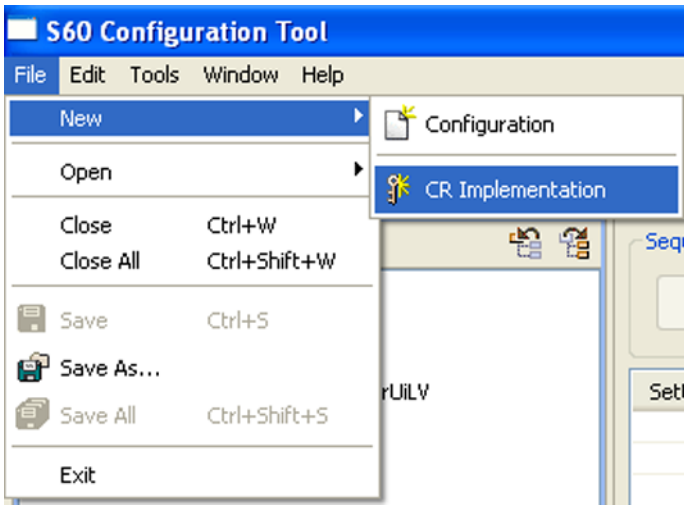
Creating a new CR implementation file

1 To create a new CR implementation file, first switch the perspective to **CenRep Definition**. This is done by selecting the perspective in the right upper corner of the main window.



You can also select the perspective from the menu: **Window** --> **Open Perspective**. **Open Perspective** -dialog opens from which you can select the correct perspective.

2 Select **File** --> **New** --> **CR Implementation**.



A **New Repository** dialog opens.

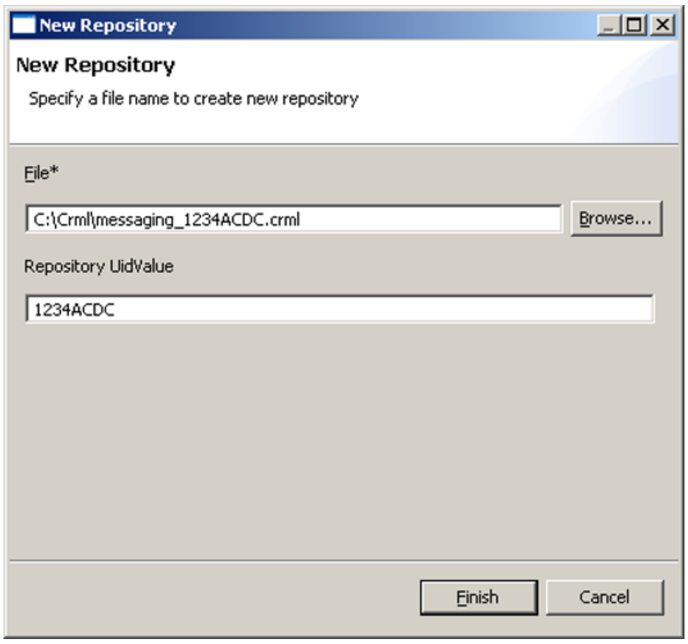
3 Enter the file path and name in the **File** text field. The file extension must be .crml and the whole directory/file path needs to be added.

Or

Select **Browse...** to open the file dialog and navigate to the desired folder. Type the name of the new CR implementation in the **File Name** text field and select **Open**.

> **Note:** The file name should contain only lowercase letters and no spaces. Normally the file name includes UidValue. For example, activeidle_10207467.crml. Note that UidValue can also have capitals from A to F.

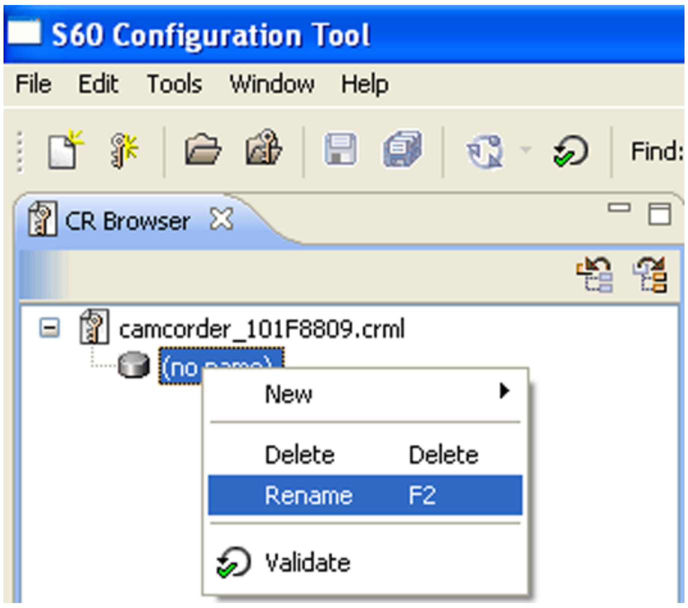
4 In the **New Repository** dialog, enter a UidValue for the repository.



5 When both fields are filled, select **Finish**.

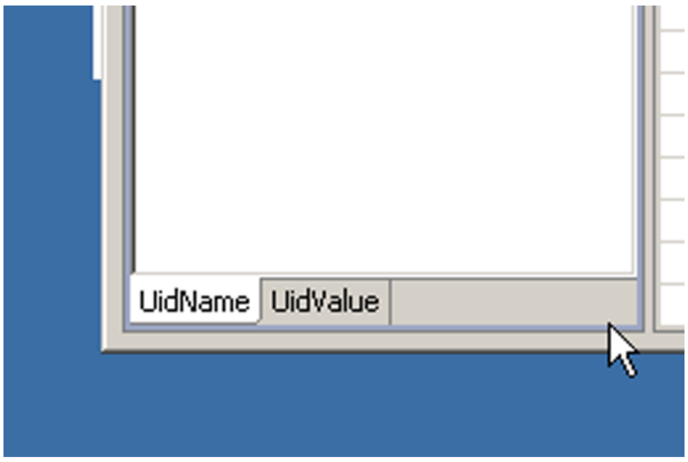
A new CR implementation file is created and visible in the **CR Browser** view.

6 Right-click on the **UidName** field currently named "(no name)" and select **Rename** from the popup menu. Give the correct name for this Uid.



7 Now you can add a new **Simple Key**, **Bitmask Key** or **Range Key**.

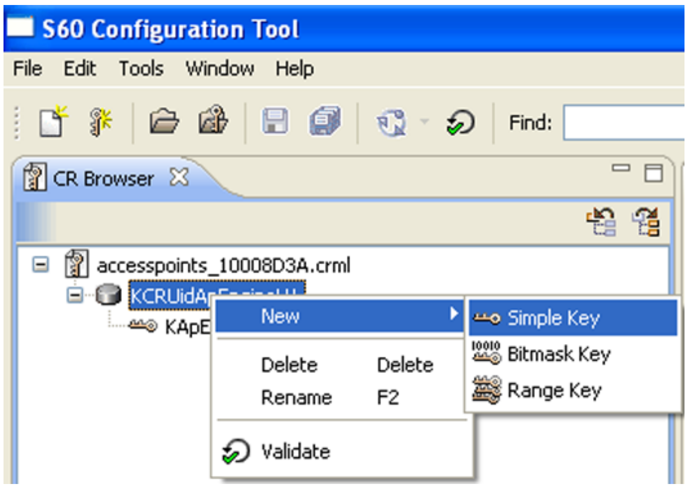
8 You can view the CR implementation files by **UidName** or **UidValue**. Select the corresponding tab at the bottom of the **CR Browser** to change the viewing mode.



Adding a new simple key

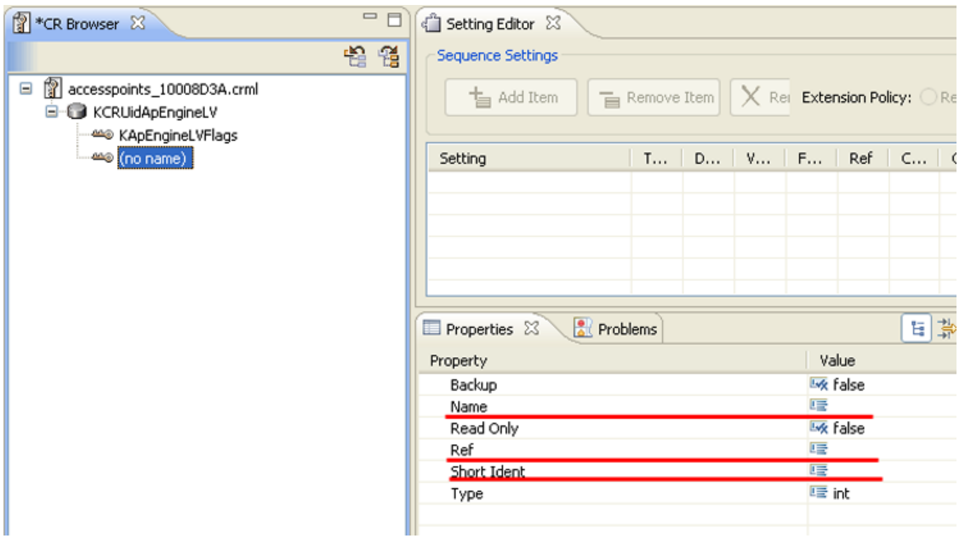
1 To add a new simple key to CR implementation, right-click on the repository.

2 A popup menu opens from which you can select **New --> Simple Key**.



3 In the **Properties** view, give values to key **Name**, **Ref** and **Short Ident** (marked with red color).

4 Change **Type**, **Backup** and **Read Only** if necessary.



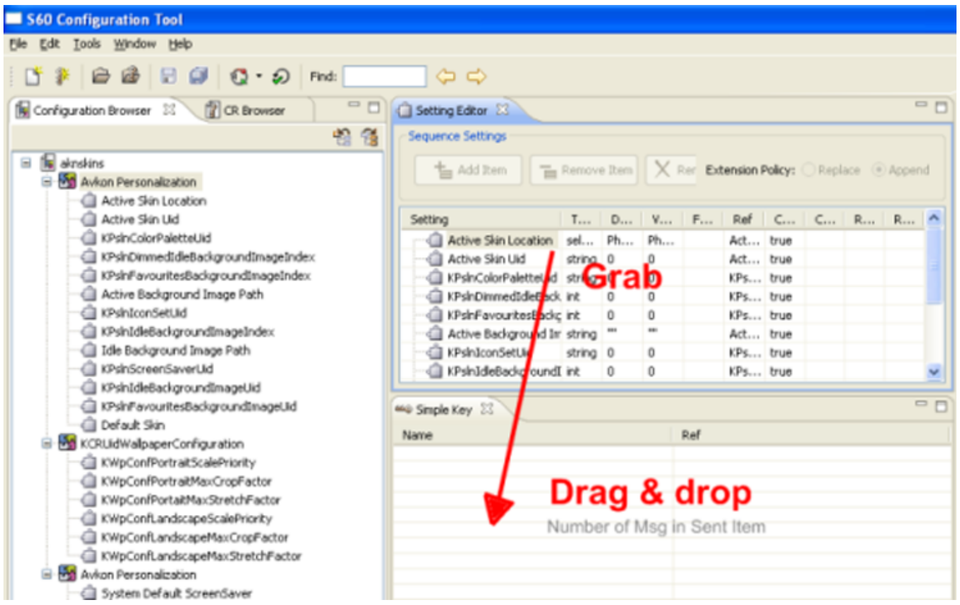
Property	Value
Backup	false
Name	
Read Only	false
Ref	
Short Ident	
Type	int

This is how you can create a new key from scratch. However, if you want to create a key for an existing setting, follow the instructions below:

1 To create a key to an existing setting, open an existing configuration by selecting **File** -> **Open** -> **Configuration**.

2 Use the **File** dialog to browse to the desired confml file and select **Open**. Now you should be able to see both **Setting Editor** and **Simple Key** views.

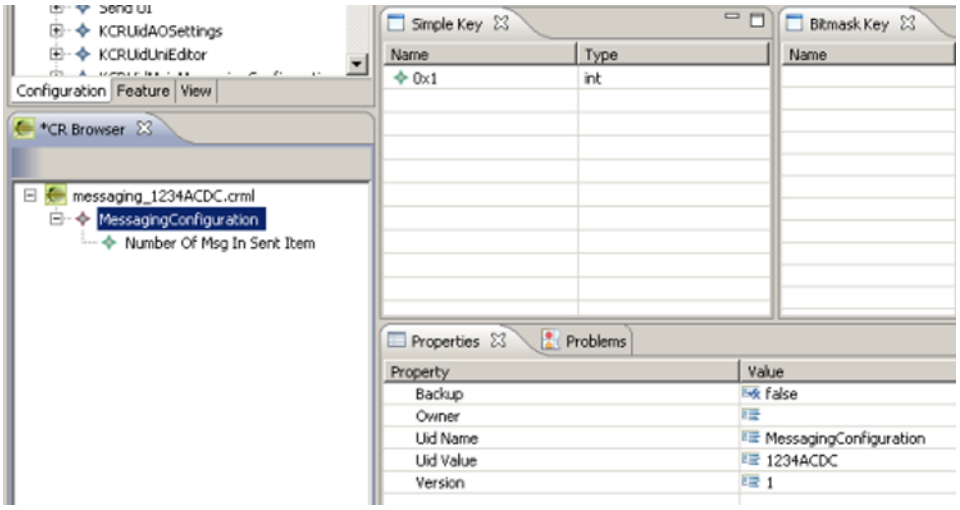
3 Make sure that either **UidName** or **UidValue** is selected from the CR Browser. Then select a feature from the Configuration Browser so that you can see the desired setting in the Setting Editor. Now you can drag and drop the setting to the **Simple Key** view.



akrskins
- Avkon Personalization
  - Active Skin Location
  - Active Skin Uid
  - KPslnColorPaletteUid
  - KPslnDimmedIdleBackgroundImageIndex
  - KPslnFavouritesBackgroundImageIndex
  - Active Background Image Path
  - KPslnIconSetUid
  - KPslnIdleBackgroundImageIndex
  - Idle Background Image Path
  - KPslnScreenSaverUid
  - KPslnIdleBackgroundImageUid
  - KPslnFavouritesBackgroundImageUid
  - Default Skin
- KCRUidWallpaperConfiguration
  - KWpConfPortraitScalePriority
  - KWpConfPortraitMaxCropFactor
  - KWpConfPortaitMaxStretchFactor
  - KWpConfLandscapeScalePriority
  - KWpConfLandscapeMaxCropFactor
  - KWpConfLandscapeMaxStretchFactor
- Avkon Personalization
  - System Default ScreenSaver

Setting	T...	D...	V...	F...	Ref	C...	C...	R...	R...
Active Skin Location	sel...	Ph...	Ph...		Act...	true			
Active Skin Uid	string	0	0		Act...	true			
KPslnColorPaletteUid	string	0	0		KPs...	true			
KPslnDimmedIdleBack	int	0	0		KPs...	true			
KPslnFavouritesBackg	int	0	0		KPs...	true			
Active Background Im	string	""	""		Act...	true			
KPslnIconSetUid	string	0	0		KPs...	true			
KPslnIdleBackgroundI	int	0	0		KPs...	true			

This creates a new key that has the same name as the setting and a correct **Ref**. Also **Short Ident** is automatically set to the next available value.



- Send UI
- KCRUidAOSettings
- KCRUidUniEditor

- messaging_1234ACDC.crml
  - MessagingConfiguration
    - Number Of Msg In Sent Item

Name	Type
0x1	int

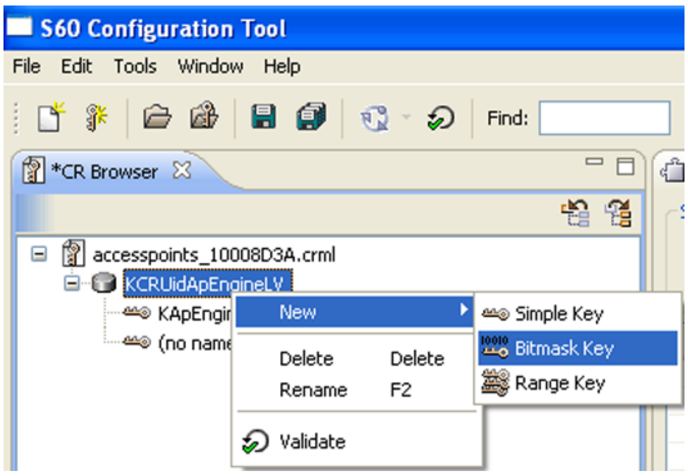
Property	Value
Backup	false
Owner	
Uid Name	MessagingConfiguration
Uid Value	1234ACDC
Version	1

Adding a new bitmask key

This is for boolean settings.

1 To add a new bitmask key, right-click on **UidName** or **UidValue**.

2 Select **New** --> **Bitmask Key** from the menu.



3 Go to the **Properties** view and give values to key **Name**, **Ref** and **Short Ident**.

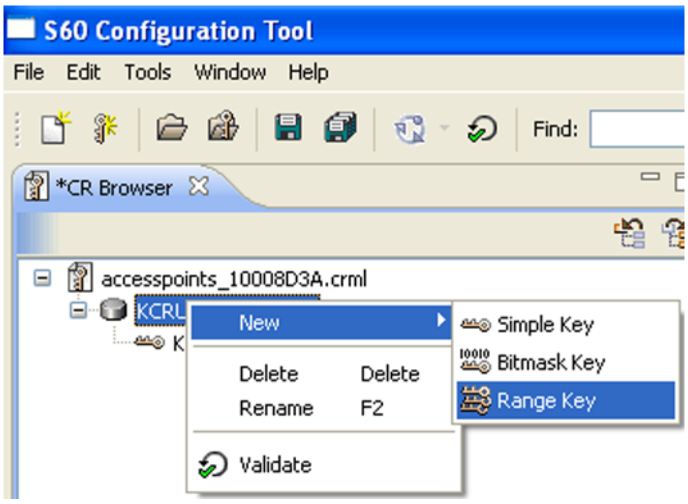
4 Drag and drop the boolean settings to the **Bitmask Key** view.

Adding a new range key

This is for sequence settings.

1 To add a new range key to a CR implementation, right-click on **UidName** or **UidValue**.

2 Select **New** --> **Range Key** from the menu.



3 Go to the **Properties** view and give values to key **Name**, **Ref**, **First Int**, **Last Int**, **Index Bits** and **First Index**.

4 Change **Backup** and **Read Only** if necessary.

5 To add keys to a range, open an existing configuration by selecting **File** --> **Open** --> **Configuration**.

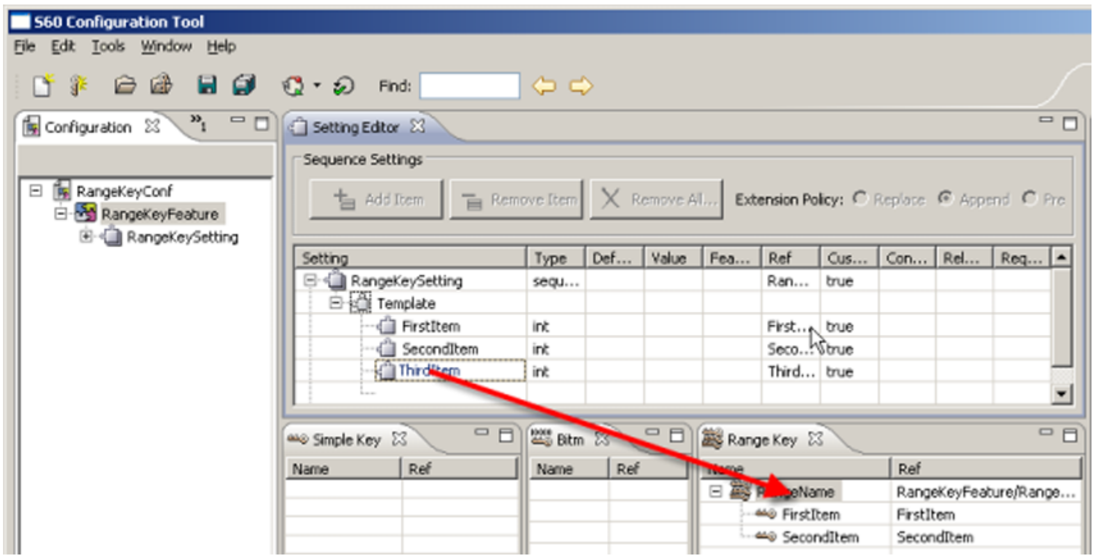
6 Use the **File** dialog to browse to the desired confml file and select **Open**.

Now you should be able to see both **Setting Editor** and **Simple Key** views.

7 Make sure that either **UidName** or **UidValue** is selected from CR Browser.

8 Select a feature from the Configuration Browser so that you can see the desired setting in the Setting Editor. Now you can drag and drop a sequence setting template item to the **Range Key** view range.

9 This creates a new key that has the same name as the setting and a correct **Ref** for both parent range and key. Drag and drop on top of an existing key to replace it.



Setting	Type	Def...	Value	Fea...	Ref	Cus...	Con...	Rel...	Req...
RangeKeySetting	sequ...				Ran...	true			
Template									
FirstItem	int				First...	true			
SecondItem	int				Seco...	true			
ThirdItem	int				Third...	true			

Name	Ref
RangeName	RangeKeyFeature/Range...
FirstItem	FirstItem
SecondItem	SecondItem

Note: It is important where you drop the item. Depending on the spot it creates either a new key or a new bit.

Saving the changes

1 To save the changes you have made to the new CR implementation file, select **File --> Save** (or use shortcut Ctrl+S).

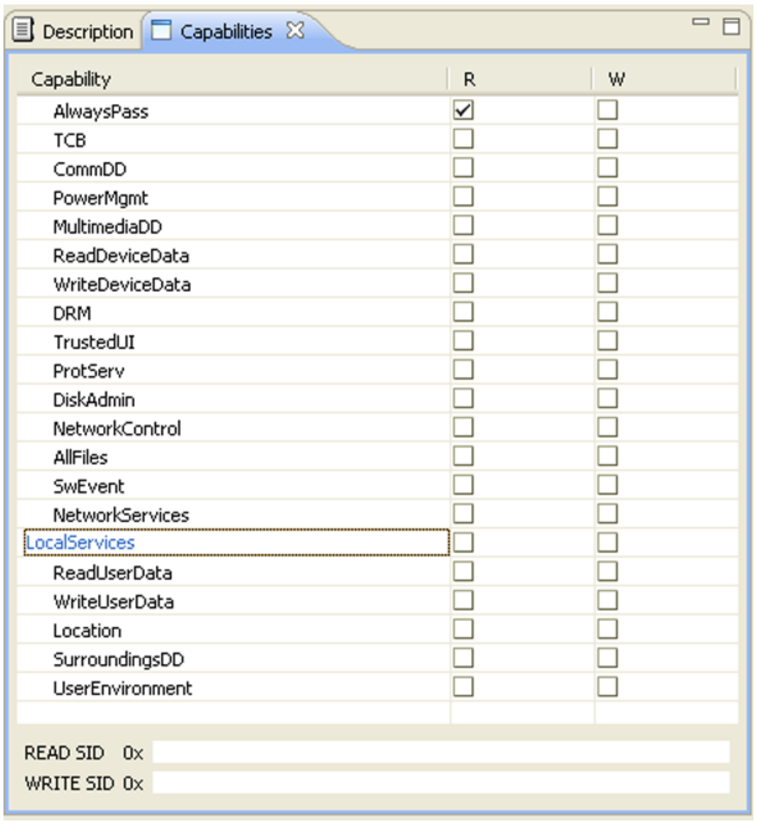
Setting the capabilities

Capabilities can be set for a key or a repository.

1 To set the capabilities, first open the **Capabilities** view (bottom-right corner in the **CenRep** perspective).

2 Select the repository or key whose capability is to be set.

3 Check/uncheck the required capability from the list. Notice that the table allows you to change the read and write capabilities. Read and write SID values can also be set within the **Capabilities** view.



Capability	R	W
AlwaysPass	✓	☐
TCB	☐	☐
CommDD	☐	☐
PowerMgmt	☐	☐
MultimediaDD	☐	☐
ReadDeviceData	☐	☐
WriteDeviceData	☐	☐
DRM	☐	☐
TrustedUI	☐	☐
ProtServ	☐	☐
DiskAdmin	☐	☐
NetworkControl	☐	☐
AllFiles	☐	☐
SwEvent	☐	☐
NetworkServices	☐	☐
LocalServices	☐	☐
ReadUserData	☐	☐
WriteUserData	☐	☐
Location	☐	☐
SurroundingsDD	☐	☐
UserEnvironment	☐	☐

READ SID 0x

WRITE SID 0x

4.3 Creating a new genconfml file

Creating a gcfml file

Genconfml files are stored in the tool's genconfml directory and they must have file extension .gcfml. Gcfml files are only read on tool startup so changes to them only take effect after tool restart. Use a text editor to create the file and use the code below as template.

```
<file xmlns="http://www.s60.com/xml/genconfml/1" name="FileName.xml(1)">
    <setting ref="Feature/Setting(2)"/>
    <setting ref="Feature/Setting(2)"/>
    <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/
Transform" xmlns:xi="http://www.w3.
    org/2001/xinclude">
        <xsl:output method="xml(3)" indent="yes" encoding="UTF-8"/>
        <xsl:template match="configuration/data/(4)">
            <(5)>
        </xsl:template>
    </xsl:stylesheet>
</file>
```

1 Replace FileName.xml with the output file name you need.

2 Replace Feature/Setting with the feature and setting pairs you need. Only those settings defined here are used in the transformation so multiple setting tags may be needed. Wildcards or features alone are not supported at the moment.

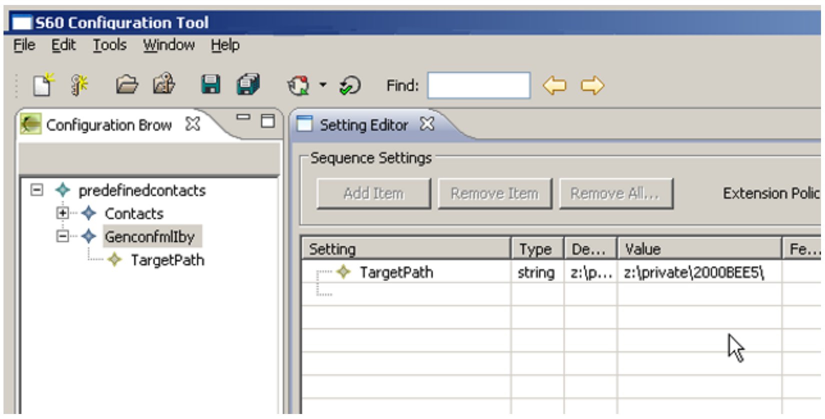
3 Define the output file type here. Possible values are xml, html and text.

4 Define the part where to apply transformation here. Root for the settings is "configuration/data".

5 Define XSLT here. Tutorial for XSLT can be found at http://www.w3schools.com/xsl/.

Defining the output file location

Feature "GenconfmlIby" with setting "TargetPath" in confml can be used to define the output file location in the IBY file. The S60 Configuration Tool does not copy output files automatically to this location.

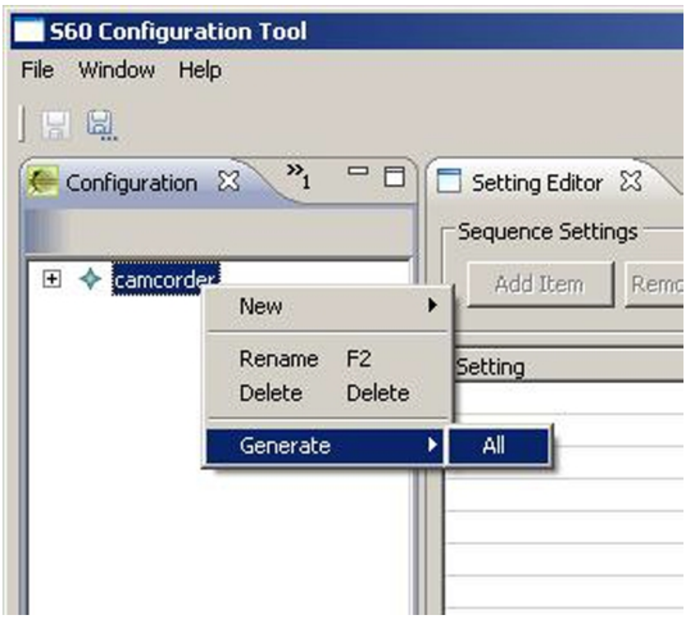


Setting	Type	De...	Value	Fe...
TargetPath	string	z:\p...	z:\private\2000BEE5\	

Running a transformation

1 To run a transformation, right-click on the configuration in the Configuration Browser and select **Generate --> All**.

Output files are generated to the result\common directory in the S60 Configuration Tool.

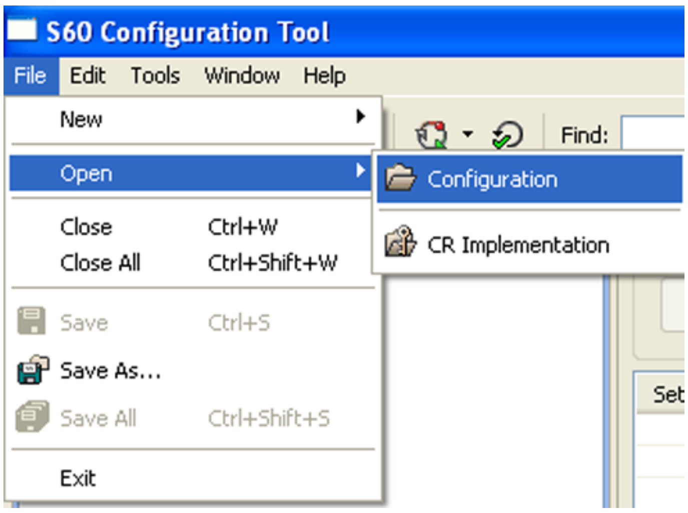


5 Generating the build

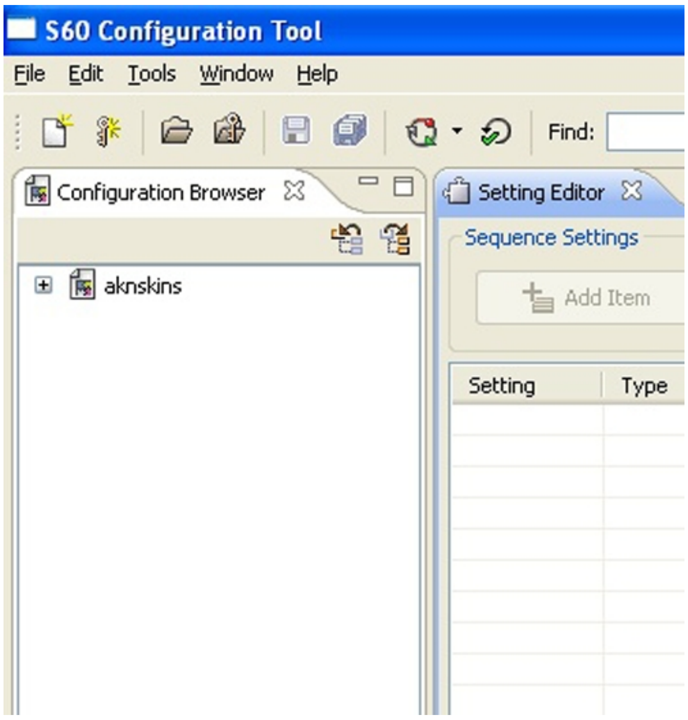
5.1 Generating an existing configuration

Opening an existing configuration

1 To open an existing configuration, select **File** --> **Open** --> **Configuration**.

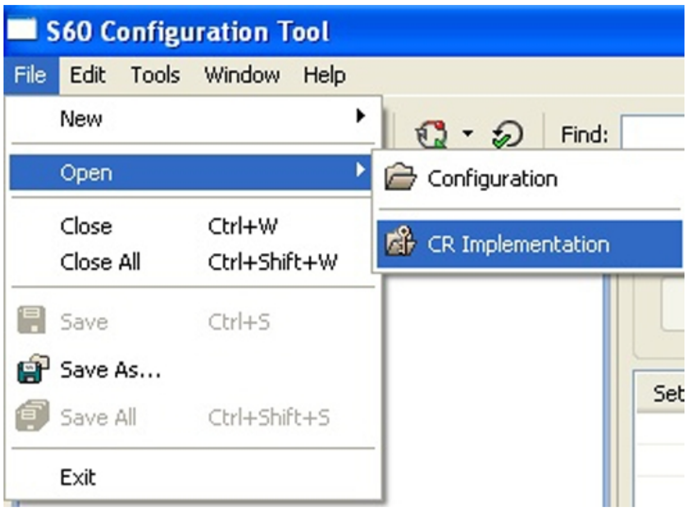


2 Select the master confml file and select **Open**. This action opens the configuration in the Configuration Browser.

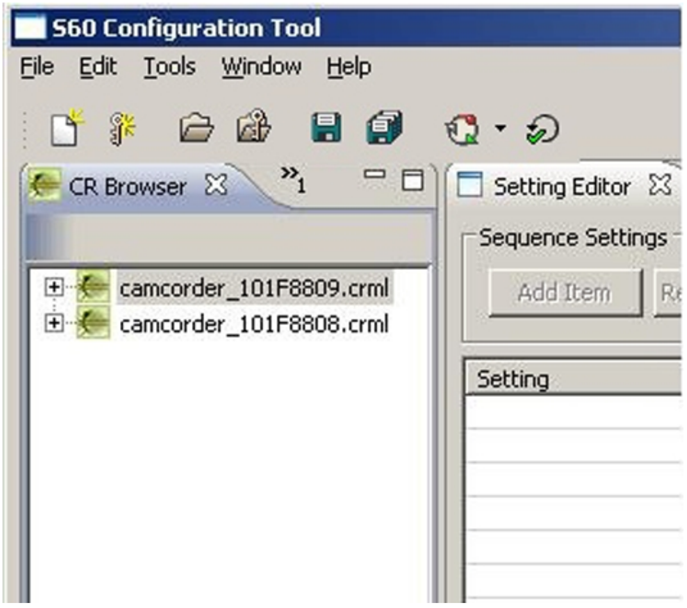


Opening the CR implementation files

1 To open the CR implementation files, select **File** --> **Open** --> **CR Implementation**.



2 Select all (Ctrl+A) crml files and select **Open**. This action opens all the CR implementation files in the CR Browser.



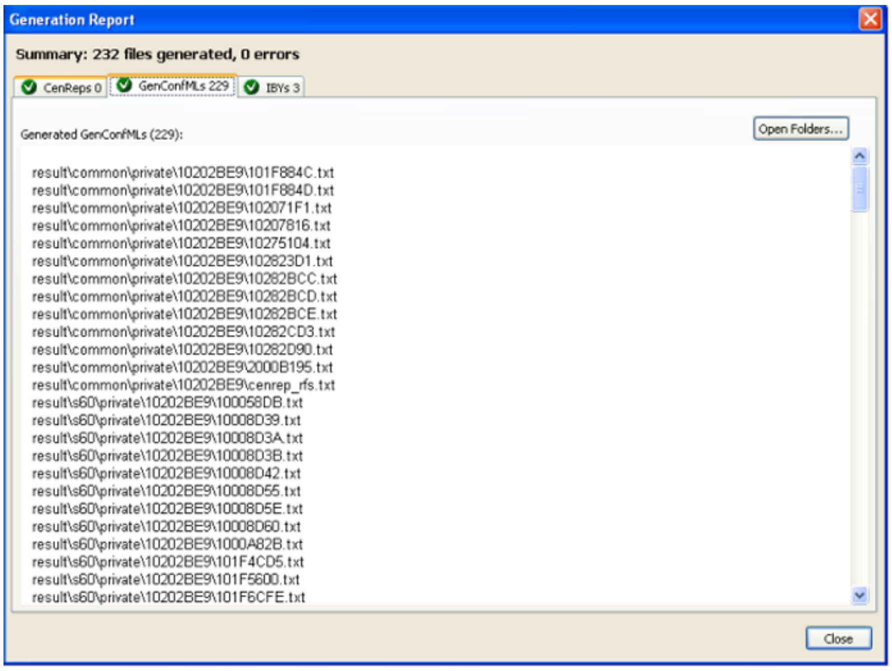
Generating variant content

1 To generate variant content, first select a root configuration.

2 Right-click on the root configuration.

3 Select **Generate --> All**.

This action generates all CR txt files, IBY files and genconfml files. Generated files are shown in the **Generation Results** dialog which also shows possible errors during generation.

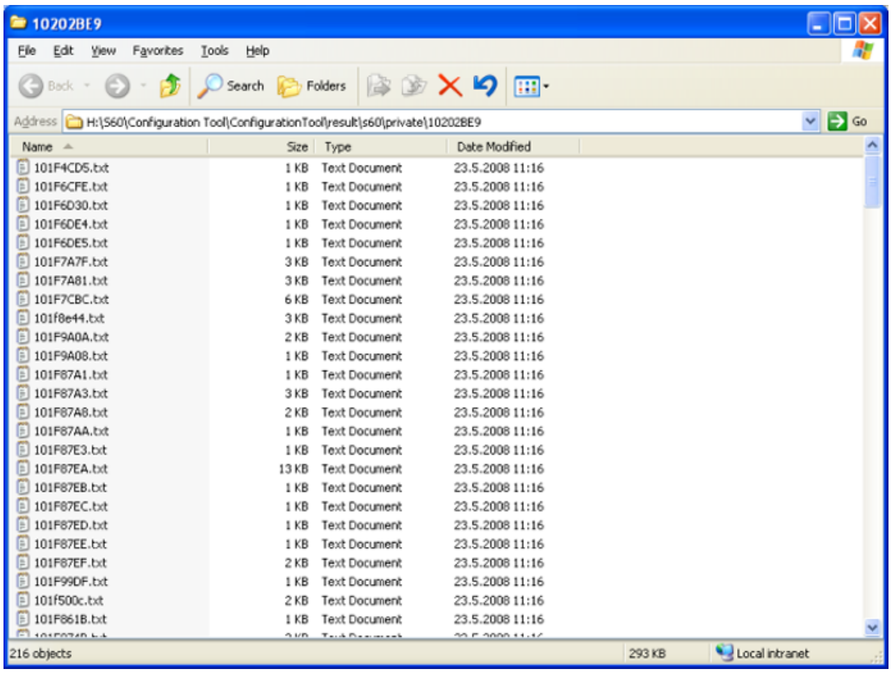


Summary: 232 files generated, 0 errors

Generated GenConfMLs (229):

```
result\common\private\10202BE9\101F884C.txt
result\common\private\10202BE9\101F884D.txt
result\common\private\10202BE9\102071F1.txt
result\common\private\10202BE9\10207816.txt
result\common\private\10202BE9\10275104.txt
result\common\private\10202BE9\102823D1.txt
result\common\private\10202BE9\10282BCC.txt
result\common\private\10202BE9\10282BCD.txt
result\common\private\10202BE9\10282BCE.txt
result\common\private\10202BE9\10282CD3.txt
result\common\private\10202BE9\10282D90.txt
result\common\private\10202BE9\2000B195.txt
result\common\private\10202BE9\cenrep_rfs.txt
result\s60\private\10202BE9\100058DB.txt
result\s60\private\10202BE9\10008D39.txt
result\s60\private\10202BE9\10008D3A.txt
result\s60\private\10202BE9\10008D3B.txt
result\s60\private\10202BE9\10008D42.txt
result\s60\private\10202BE9\10008D55.txt
result\s60\private\10202BE9\10008D5E.txt
result\s60\private\10202BE9\10008D60.txt
result\s60\private\10202BE9\1000A82B.txt
result\s60\private\10202BE9\101F4CD5.txt
result\s60\private\10202BE9\101F5600.txt
result\s60\private\10202BE9\101F6CFE.txt
```

4 Click **Open folders** to see the generated files.



Address H:\S60\Configuration Tool\ConfigurationTool\result\s60\private\10202BE9

Name	Size	Type	Date Modified
101F4CD5.txt	1 KB	Text Document	23.5.2008 11:16
101F6CFE.txt	1 KB	Text Document	23.5.2008 11:16
101F6D30.txt	1 KB	Text Document	23.5.2008 11:16
101F6DE4.txt	1 KB	Text Document	23.5.2008 11:16
101F6DE5.txt	1 KB	Text Document	23.5.2008 11:16
101F7A7F.txt	3 KB	Text Document	23.5.2008 11:16
101F7A81.txt	3 KB	Text Document	23.5.2008 11:16
101F7CBC.txt	6 KB	Text Document	23.5.2008 11:16
101f8e44.txt	3 KB	Text Document	23.5.2008 11:16
101F9A0A.txt	2 KB	Text Document	23.5.2008 11:16
101F9A08.txt	1 KB	Text Document	23.5.2008 11:16
101F87A1.txt	1 KB	Text Document	23.5.2008 11:16
101F87A3.txt	3 KB	Text Document	23.5.2008 11:16
101F87A8.txt	2 KB	Text Document	23.5.2008 11:16
101F87AA.txt	1 KB	Text Document	23.5.2008 11:16
101F87E3.txt	1 KB	Text Document	23.5.2008 11:16
101F87EA.txt	13 KB	Text Document	23.5.2008 11:16
101F87EB.txt	1 KB	Text Document	23.5.2008 11:16
101F87EC.txt	1 KB	Text Document	23.5.2008 11:16
101F87ED.txt	1 KB	Text Document	23.5.2008 11:16
101F87EE.txt	1 KB	Text Document	23.5.2008 11:16
101F87EF.txt	2 KB	Text Document	23.5.2008 11:16
101F99DF.txt	1 KB	Text Document	23.5.2008 11:16
101f500c.txt	2 KB	Text Document	23.5.2008 11:16
101F861B.txt	1 KB	Text Document	23.5.2008 11:16

216 objects 293 KB Local intranet

5.2 Generating a cenrep txt file

Opening an existing configuration

1 To open an existing configuration, select **File** --> **Open** --> **Configuration**.

2 Select an existing confml file and select **Open**.

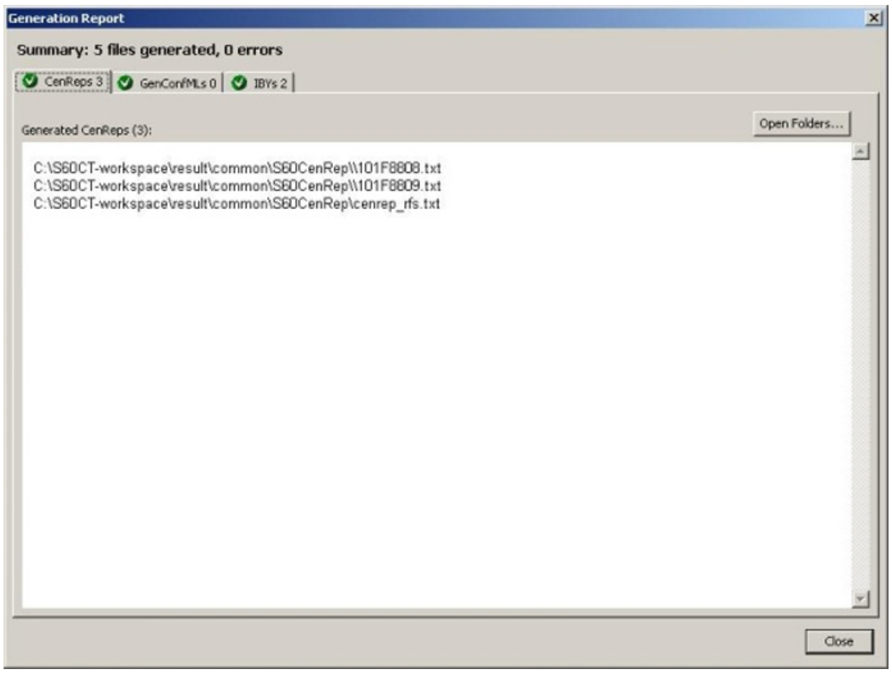
Opening implementation files

1 To open implementation files, select **File** --> **Open** --> **CR Implementation**.

2 If there are multiple implementation files, select all (Ctrl+A) crml's and select **Open**.

Generating cenrep txt files

1 To generate cenrep txt files, select a configuration.
2 Right-click on the root configuration.
3 Select **Generate --> All**. This action generates all cenrep txt files, IBY files and genconfml files. Generated files are shown in **Generation Results** dialog which also shows possible errors during generation.
4 After the tool has generated the cenrep txt files, select **Open Folders...** in the **Generation Report** dialog.



Summary: 5 files generated, 0 errors

Generated CenReps (3):

```
C:\S60CT-workspace\result\common\S60CenRep\\101F8808.txt
C:\S60CT-workspace\result\common\S60CenRep\\101F8809.txt
C:\S60CT-workspace\result\common\S60CenRep\cenrep_rfs.txt
```

This action opens the result folders in the developer machine.

Generated content file structure

In the developer use case, the result folder contains only the 'common' folder.

'Common' folder contains cenrep txt and iby files generated by the developer.

5.3 Using the command line interface for the S60 Configuration Tool

The S60 Configuration Tool command line interface (cli) offers a mechanism to generate data with scripts and integrate the tool to the build process. The command line interface is called via the cli.cmd command. For this the S60 Configuration Tool installation must be in the path.

Command cli_build.cmd can be used to generate the master confml file before generating the cenreps. The generated master file will include all confml found from the confml directory (specified with argument -confml). Before generating, the tool validates all files and if any errors occur, the cenreps are not generated. Validation errors can be ignored with argument -ignore_errors. In this case the tool generates cenreps even if errors occur.

Command line arguments for cli.cmd

cli.cmd -master <file> [-vds|-vdc [-save] <path to VariantData.xml>] -impl <dirs> -output <dir> -ibyml <files|dirs> -report <file> -ignore_errors

The following table lists the command line arguments for cli.cmd:

`-master <file>`	File name and path of the master confml file. **Mandatory argument**
`-impl <dirs>`	Directories that contain implementation crml files separated by semicolons (;). **Mandatory argument**
`-output <dir>`	Output-folder of the cenreps. This overrides the location value in ibyml files.
`-ibyml <files|dirs>`	Ibyml files or directories separated by semicolon (;). If not given, ibyml file is expected to be in impl dir.
`-vds <file>`	File name and path of the VariantData.xml file. Import single VariantData. Values are applied but not saved.
`-vdc <file>`	File name and path of the VariantData.xml file. Import chain of VariantDatas. Values are applied but not saved.
`-save`	Apply and save VariantData values.
`-report <file>`	File name and path of the report file. This file contains name and full path of every generated cenrep file.
`-ignore_errors`	Ignores validation errors and generates cenreps even if errors occur.

Command line arguments for cli_build.cmd

cli_build.cmd -master_conf <name> -impl <dir> -confml <dir> -iby <dir> -cenrep <dir> -extra_confml <dir> -report <file> -ignore_errors

The following table lists the command line arguments for cli_build.cmd:

`-master_conf <name>`	Name of the master confml file that is generated (for example S60). **Mandatory argument**
`-impl <dirs>`	Directories that contain implementation crml files separated by semicolons (;). **Mandatory argument**
`-confml <dir>`	A directory that contains confml files. Master confml is generated to this directory. **Mandatory argument**
`-iby <dir>`	Output directory of the iby files. If not specified, default directory \epoc32\rom\inlcude will be used.
`-no_iby`	Do not generate iby files.
`-cenrep <dir>`	Output directory of the cenrep files. If not specified, cenreps will be generated to three default directories: \epoc32\data\z\private\10202BE9 \epoc32\RELEASE\winscw\UDEB\Z\private\10202BE9 \epoc32\RELEASE\winscw\UREL\Z\private\10202BE9
`-extra_confml <dir>`	A directory that contains additional confml files. All confml files from this directory and all subdirectories are copied to <confml> directory. **Optional argument**
`-report <file>`	File name and path of the report file. This file contains name and full path of every generated cenrep file.
`-ignore_errors`	Ignores validation errors and generates cenreps even if errors occur.

5.4 Creating a delta rom image from operator variant files (ROFS3)

Creating a delta rom image

You must have the correct version of the S60 Configuration Tool installed. You should also have the existing confml and crml files.

1. To create a delta rom image, you must first create cenrep files from confml/crml files
   - Cli-master path to the S60Master-folder - crml path to the Crml-folder - ibyml path to the Ibym-folder - output to the Output-folder
   - The output is generated by default to normal cenrep and iby locations
2. Copy a variant creation template for imaker to the system (for example \epoc32\rom\config\)
   - image_conf_operatorvariant.mk
3. Go to the Output-folder
4. Call imaker
   - imaker -f image_conf_operatorvariant.mk -f pathtoproduct.mk rofs3
   - For example, imaker -f \epoc32\rom\config\image_conf_operatorvariant.mk -f \epoc32\rom \config\<family>\<product>\image_conf_<product>_ui.mk rofs3

The image is created with the S60 Configuration Tool generated cenrep files.

6 Glossary

Term or abbreviation	**Definition**
Bitmask key	A key the value of which is a bitmask. Bits of the key value are meaningful and can be changed independently. There is a boolean setting for each significant bit of the bitmask.
Capability	Access level to a key (repository) value. Defines necessary conditions that should be satisfied for an app to access a key (repository) value. Repository access and its capabilities define default access for all keys within the repository.
Central repository	Symbian OS service that provides a user interface to allow one or more clients to open repositories, and to provision and retrieve information from those repositories.
Configuration	Specific values for a collection of settings. Used to configure terminal SW for certain platform release, product or trade customer (for example operator). In Configuration ML is the root element of the language. It also includes feature and setting definitions.
ConfML file	ConfML (S60 Configuration ML) is an XML-based markup language (i.e. an XML application) for defining SW configuration elements and configuration data of S60 terminals.
Crml file	Crml (Central Repository ML) is used for settings stored at runtime in Symbian Central Repository component.
Extension policy	A mechanism for controlling how sequence item-settings for a single setting in multiple configurations are handled.
Feature	A collection of related settings.
Genconfml file	Generic Configuration File ML. Used for settings stored at runtime in arbitrary text format. The file is the root element of the language. Each generated file must be defined in its own Generic Configuration File XML file.
Generated CenRep file	A system text file which is converted to a binary during the build process and used by the phone's applications when loaded into the phone.
ML	Markup language. A markup language is an artificial language using a set of annotations to text that describe how text is to be structured, laid out, or formatted.
Option	Allowed value for a setting.
Perspective	A perspective defines the set and layout of views in the tool. They also control what is shown in certain menus and toolbars.
Range key	Predefines a range of keys that can be filled with keys during configuration, if the range is configurable. Range key takes values from a set of settings usually organized into a sequence setting.
Read & write SID	When these are defined, only the client with matching SID can have access to the key when it also has all the listed capabilities.

Reference	For example, setting ref, feature ref. These are absolute identifiers of a setting. Each configuration defines zero or more configuration elements in terms of features and settings. A feature groups together related settings. One setting describes a certain configuration capability implemented in software. Every feature must be identified uniquely by using a ref attribute.
S60 Configuration Tool	S60 specific tool to configuring SW build (for example variant) based on the configuration planned from S60 release 3.2.2 onwards.
Setting	Defines a single configurable element (for example mailbox name).
Sequence setting	The data of a sequence type setting represents an ordered collection (i.e. a list) of item-settings that each consists of one or more sub-settings. Each item-setting in the sequence must have exactly same set of sub-settings. The sub-settings can be of any data-type, except sequence. Therefore sequence of sequences is not allowed . The sub-settings of sequence are defined using child setting elements of the sequence setting.
Simple key	A single value key. The key presents a single configuration point. Simple key takes value from one setting only (confml).
Validation	A mechanism for checking semantic and logic correctness of confml/crml files.
Value	Defines a value of a setting. Single customisable value of a certain setting (for example mailbox name).
Variant	A specific kind of configuration.
View	For rearranging settings into new structure and redefining their properties regardless of how they were originally defined.