

# PKIFv2 Resources Integration Guide

## Introduction

The PKIF public key enablement library provides a set of graphical interfaces to enable easy configuration of parameters that govern PKI-related processing. These interfaces can be integrated into applications providing a consistent configuration interface and creating opportunities for shared configurations. This document describes how to integrate the resources and serialization libraries into an application.

Two libraries are provided to enable integration of a common user interface for configuring PKIF settings and to serialize those settings: PKIFResources.dll and PKIFConfigSerialization.dll. PKIFResources.dll provides two notebook resources that can be used to

## Serialization and Configuration Basics

The PKIF serialization library provides means of serializing and de-serializing mediator/colleague sets and **CPKIFPathSettings** objects. Similarly, the PKIF resources library provides graphical means of configuring mediator/colleague sets and **CPKIFPathSettings** objects. There are four primary classes that work in concert to support configuration and serialization of these objects:

- **CPKIFMediatorSerializer**
- **CPKIFPathSettingsSerializer**
- **CPKIFPkiEnvironmentDefinitionNotebook**
- **CPKIFPathSettingsNotebook**

The **CPKIFMediatorSerializer** and **CPKIFPkiEnvironmentDefinitionNotebook** classes consume and produce **IPKIFMediatorPtr** objects that contain a set of mediator and colleague objects that define the PKI environment for a particular application.

The **CPKIFPathSettingsSerializer** and **CPKIFPathSettingsNotebook** classes consume and produce **CPKIFPathSettingsPtr** objects that contain parameters that govern certification path processing operations, including the standard RFC3280 path validation input parameters.

## Mediator/Colleague Serialization and Configuration

The following code snip demonstrates the usage of **CPKIFMediatorSerializer** and **CPKIFPkiEnvironmentDefinitionNotebook** to load previously saved settings, enable configuration of those settings followed by saving the new settings.

```
CPKIFMediatorSerializer ms;  
IPKIFMediatorPtr m = ms.LoadSettings(
```

```

    "SomeApplication", "SomeCompany");

CPKIFPkiEnvironmentDefinitionNotebook pedn(NULL,-1,
    wxT("PKI Environment Definition"),
    wxDefaultPosition,wxDefaultSize, wxDEFAULT_DIALOG_STYLE);
pedn.SetInitialMediator(m);
pedn.Center();
if(wxID_OK == pedn.ShowModal())
{
    m = pedn.GetMediator();
    ms.SaveSettings(m, "SomeApplication", "SomeCompany");
}

```

## Path Settings Serialization and Configuration

The following code snip demonstrates the usage of **CPKIFPathSettingsSerializer** and **CPKIFPathSettingsNotebook** to load previously saved settings, enable configuration of those settings followed by saving the new settings.

```

CPKIFPathSettingsSerializer pss;
CPKIFPathSettingsPtr ps;
ps = pss.LoadSettings("SomeApplication", "SomeCompany");

CPKIFPathSettingsNotebook psn(NULL, -1,
    wxT("Path Settings"), wxDefaultPosition, wxDefaultSize,
    wxDEFAULT_DIALOG_STYLE);
psn.SetInitialPathSettings(ps);
psn.Center();
if(wxID_OK == psn.ShowModal())
{
    ps = psn.GetPathSettings();
    pss.SaveSettings(ps, "SomeApplication", "SomeCompany");
}

```