Bluetooth Engine API

Note: This API is not part of the public SDK. It can be found in the SDK API Plug-in.

APIPurpose

The main purpose of Bluetooth Engine is to provide a higher-level abstraction API for Bluetooth applications and services.The Bluetooth Engine API provides access to MCM settings, Bluetooth device configurations and Bluetooth connection settings.

Header Files:

```
#include <bteng.h>
#include <bttypes.h>
#include <btdevice.h>
```

Link against:

```
LIBRARY   bteng.lib  bluetooth.lib  btdevice.lib
```

CAPABILITY:

```
CAPABILITY   LocalServices WriteDeviceData //for Set API's
```

Use cases

-Get/Set the value of Bluetooth MCM settings in this device. The Bluetooth MCM settings include the Bluetooth power state, discoverability mode, search mode, local Bluetooth name and Bluetooth SAP enable state

-Get a list of the paired Bluetooth devices stored in the local Bluetooth registry.Bluetooth Engine retrieves the devices via Bluetooth registry client API. The list of devices will be ordered by locale standard collation.

-It adds/retrieves/modifies/deletes a device(s) into/from Registry and also Sets the device security.

Example code

1)Derive your class from MBTMCMSettingsCB and implement the functions in them.

2)Create an Object of CBTMCMSettings.

```
iBtSettings = CBTMCMSettings::NewL( this );
```

-Get All Settings:

```
TBool powerState;
TBTDiscoverabilityMode discoverabilityMode;
TBTSearchMode searchMode;
TBuf<50> localName;
TBool nameModifiedFlag;
TBTSapMode sapMode;
iBtSettings->GetAllSettings(powerState,discoverabilityMode,searchMode,
localName,nameModifiedFlag,sapMode);
```

-Get local Bluetooth Name

```
TBuf<50> localName;
iBtSettings->GetLocalBTName(localName);
CEikonEnv::InfoWinL(_L("BT Name"),localName);
```

-Set local Bluetooth Name

```
TBuf<50> localName;
CAknTextQueryDialog* dlg = CAknTextQueryDialog::NewL(localName);
CleanupStack::PushL(dlg);
TInt err = dlg->ExecuteLD(R_DATA_QUERY_DIALOG);
if(err)
  {
  iBtSettings->SetLocalBTName(localName);
  }
else
  {
  CEikonEnv::InfoWinL(_L("name not Entered"),_L(""));
  }
```


(C) Copyright Nokia 2013. All rights reserved.

```
CleanupStack::Pop(dlg);
```

-Get Discoverability Mode

```
TBTDiscoverabilityMode discoverabilityMode;
iBtSettings->GetDiscoverabilityMode(discoverabilityMode);
if(discoverabilityMode == EBTDiscModeHidden)
  {
  CEikonEnv::InfoWinL(_L("BlueTooth Discoverability"),_L("Hidden"));
  }
  else if(discoverabilityMode == EBTDiscModeGeneral)
       {
       CEikonEnv::InfoWinL(_L("BlueTooth Discoverability"),_L("General"));
       }
       else
       {
       CEikonEnv::InfoWinL(_L("BlueTooth Discoverability"),_L("Temporary"));
       }
```

-Set Discoverability Mode

```
TBTDiscoverabilityMode discoverabilityMode(EBTDiscModeHidden);
//Can be EBTDiscModeGeneral or EBTDiscModeTemporary
iBtSettings->SetDiscoverabilityModeL(discoverabilityMode);
```

-Get Search Mode

```
TBTSearchMode searchMode;
iBtSettings->GetSearchMode(searchMode);
if(searchMode ==EBTSearchModeGeneral)
  {
  CEikonEnv::InfoWinL(_L("BlueTooth SearchMode"),_L("General"));
  }
else
  {
  CEikonEnv::InfoWinL(_L("BlueTooth SearchMode"),_L("Limited"));
  }
```

-Set Search Mode

```
TBTSearchMode searchMode(EBTSearchModeGeneral);//Can be EBTSearchModeLimited
iBtSettings->SetSearchMode(searchMode);
```

-Get Power State

```
TBool powerState;
iBtSettings->GetPowerState(powerState);
if(powerState)
  {
  CEikonEnv::InfoWinL(_L("BlueTooth On"),_L(""));
  }
else
  {
  CEikonEnv::InfoWinL(_L("BlueTooth Off"),_L(""));
  }
```

-Set Power State

```
iBtSettings->SetPowerState(ETrue);
//ETrue for Switching On, EFalse for Switching Off.
```

-Get BTSap Mode

```
TBTSapMode sapMode;
iBtSettings->GetBTSapEnableState(sapMode);
if(sapMode == EBTSapDisabled)
  {
  CEikonEnv::InfoWinL(_L("BlueTooth SAPMode"),_L("Disabled"));
  }
else
  {
  CEikonEnv::InfoWinL(_L("BlueTooth SAPMode"),_L("Enabled"));
  }
```

-Set BTSap Mode

```
TBTSapMode sapMode(EBTSapEnabled);//EBTSapDisabled
iBtSettings->EnableBTSap(sapMode);
```

Bluetooth Address of a device can be obtained by *#2820# star-hash code

```
RBTiDeviceHandler iDeviceHandler;
```

-Get List of paired devices

```
RBTDeviceHandler deviceHandler;
CBTDeviceArray* selectedDeviceArray = new (ELeave) CArrayPtrFlat<CBTDevice>(4);
CleanupStack::PushL(selectedDeviceArray);
TInt err =  deviceHandler.GetPairedDevicesL(*selectedDeviceArray);
if(err == KErrNone)
    {
    TBuf<50> deviceName;
```


(C) Copyright Nokia 2013. All rights reserved.

```
    for(TInt i = 0 ; i<selectedDeviceArray->Count();i++)
 {
 deviceName.Copy(selectedDeviceArray->At(i)->DeviceName());
 CEikonEnv::InfoWinL(_L("Device"),deviceName);
 }
    }
else
    {
    TBuf<50> errBuf;
    errBuf.AppendNum(err);
    CEikonEnv::InfoWinL(_L("Err"),errBuf);
    }
CleanupStack::PopAndDestroy(selectedDeviceArray);
```

-Add a Device into Registry

```
//000000000000 is Bluetooth Device Address in decimal which
// is unique to a device

TInt64 lBTIntAddr = 000000000000;
TBTDevAddr lBTDevAddr(lBTIntAddr);
CBTDevice* lBTDevice = CBTDevice::NewL(lBTDevAddr);
TInt lErr = iDeviceHandler.AddDeviceIntoRegistryL(*lBTDevice);
TBuf<50> errBuf;
errBuf.AppendNum(lErr);

if ( lErr == KErrNone )
   {
   CEikonEnv::InfoWinL(_L("added to registry"),errBuf);
   }
else
   {
   CEikonEnv::InfoWinL(_L("Err"),errBuf);
   }
```

-Get a Device from the Registry

```
//000000000000 is Bluetooth Device Address in decimal which
// is unique to a device

TInt64 lBTIntAddr = 000000000000;
TBTDevAddr lBTDevAddr(lBTIntAddr);
CBTDevice* lBTDevice = CBTDevice::NewL();
TInt lErr = iDeviceHandler.GetDeviceFromRegistryL(*lBTDevice, lBTDevAddr);
if ( lErr == KErrNone )
   {
   // using lBTDevice, one can retrieve DeviceName,
   // its FriendlyName and its Bluetooth Address

   //sample code for retrieving DeviceName

   TBuf8<20> lDeviceName8 = lBTDevice->DeviceName();
   TBuf<20> lDeviceName16;
   lDeviceName16.Copy(lDeviceName8);

   EikonEnv::InfoWinL(_L("Device Name"), lDeviceName16);
   }
else
   {
   TBuf<50> errBuf;
   errBuf.AppendNum(lErr);
   CEikonEnv::InfoWinL(_L("Err"),errBuf);
   }
```

-Getting Devices from Registry

```
CBTDeviceArray* lSelectedDeviceArray = new (ELeave)
                                 CArrayPtrFlat<CBTDevice>(10);
CleanupStack::PushL(lSelectedDeviceArray);
TInt err = iDeviceHandler.GetDevicesFromRegistryL(*lSelectedDeviceArray);
if(err == KErrNone)
   {
   TBuf<50> deviceName;
   for(TInt i = 0 ; i<lSelectedDeviceArray->Count();i++)
 {
       deviceName.Copy(lSelectedDeviceArray->At(i)->DeviceName());
 CEikonEnv::InfoWinL(_L("Device"),deviceName);
 }
   }
else
   {
   TBuf<50> errBuf;
   errBuf.AppendNum(err);
   CEikonEnv::InfoWinL(_L("Err"),errBuf);
   }
CleanupStack::PopAndDestroy(lSelectedDeviceArray);
```

-Delete a Device from Registry

```
//000000000000 is Bluetooth Device Address in decimal which
// is unique to a device

TInt64 lBTIntAddr = 000000000000;
TBTDevAddr lBTDevAddr(lBTIntAddr);
CBTDevice* lBTDevice = CBTDevice::NewL(lBTDevAddr);
TInt lErrDel = iDeviceHandler.DeleteDeviceFromRegistryL(*lBTDevice);

if ( lErrDel == KErrNone )
   {
   CEikonEnv::InfoWinL(_L("deletion"),_L("successful"));
   }
else
   {
   TBuf<50> errBuf;
   errBuf.AppendNum(lErr);
   CEikonEnv::InfoWinL(_L("deletion Err"),errBuf);
   }
```


(C) Copyright Nokia 2013. All rights reserved.

-Modify a Device in Registry

```
//000000000000 is Bluetooth Device Address in decimal which
// is unique to a device

TInt64 lBTIntAddr = 000000000000;
TBTDevAddr lBTDevAddr(lBTIntAddr);
CBTDevice* lBTDevice = CBTDevice::NewL(lBTDevAddr);
TBuf<10> lDevFrdName(_L("XYZ"));
lBTDevice->SetFriendlyNameL(lDevFrdName);
TInt lErrMod = iDeviceHandler.ModifyDeviceInRegistryL(*lBTDevice);

if ( lErrMod == KErrNone )
   {
   CEikonEnv::InfoWinL(_L("modification"),_L("successful"));
   }
else
   {
   TBuf<50> errBuf;
   errBuf.AppendNum(lErrMod);
   CEikonEnv::InfoWinL(_L("modification Err"),errBuf);
   }
```

-Set Device Security

```
//000000000000 is Bluetooth Device Address in decimal which
// is unique to a device

TInt64 lBTIntAddr = 000000000000;
TBTDevAddr lBTDevAddr(lBTIntAddr);
CBTDevice* lBTDevice = CBTDevice::NewL(lBTDevAddr);
TInt lErr = iDeviceHandler.SetDeviceSecurityL(*lBTDevice, ETrue,ETrue,ETrue);
TBuf<50> errBuf;
errBuf.AppendNum(lErr);

if ( lErr == KErrNone )
   {
   CEikonEnv::InfoWinL(_L("security settings"),errBuf);
   }
else
   {
   CEikonEnv::InfoWinL(_L("Err"),errBuf);
   }
```

Example project

- BluetoothEngine_ExApp

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.