Archived:Controlling light settings using CHWRMLight

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Overview

This code snippet shows how the `CHWRMLight` class can be used to control light settings. The example class `CLightController` contains methods to reserve and release light targets, set light target to on, off, or blinking, and it also receives notifications about light target status changes by implementing the interface `MHWRMLightObserver`.

This snippet can be self-signed.

MMP file

The following libraries are required:

```
LIBRARY HWRMLightClient.lib
LIBRARY avkon.lib //CAknListQueryDialog CAknNumberQueryDialog
```

Resource file

```
#include <eikon.rh>
#include <avkon.rsg>
#include <avkon.rh>

RESOURCE ARRAY r_lightcontroller_bool_items
    {
  items =
    {
    LBUF { txt = "EFalse"; },
    LBUF { txt = "ETrue"; }
    };
    }

RESOURCE DIALOG r_lightcontroller_light_intensity_query
  {
  flags=EGeneralQueryFlags;
  buttons=R_AVKON_SOFTKEYS_OK_CANCEL;
    items=
    {
    DLG_LINE
      {
      type=EAknCtQuery;
      id=EGeneralQuery;
      control= AVKON_DATA_QUERY
        {
          layout = ENumberLayout;
          label = "Enter light intensity (0 to 100)";
          control = AVKON_INTEGER_EDWIN
            {
            min=0;
            max=100;
            };
        };
      }
    };
  }

RESOURCE DIALOG r_lightcontroller_duration_query
  {
  flags=EGeneralQueryFlags;
  buttons=R_AVKON_SOFTKEYS_OK_CANCEL;
    items=
    {
    DLG_LINE
      {
      type=EAknCtQuery;
      id=EGeneralQuery;
      control= AVKON_DATA_QUERY
        {
          layout = ENumberLayout;
          label = "Enter duration (0 to 2147482)";
          control = AVKON_INTEGER_EDWIN
            {
            min=0;
            max=2147482;
            };
        };
      }
    };
  }

RESOURCE ARRAY r_lightcontroller_lightstarget_items
    {
  items =
    {
    LBUF { txt = "ENoTarget"; },
```


(C) Copyright Nokia 2013. All rights reserved.

```
    LBUF { txt = "EPrimaryDisplay"; },
    LBUF { txt = "EPrimaryKeyboard"; },
    LBUF { txt = "EPrimaryDisplayAndKeyboard"; },
    LBUF { txt = "ESecondaryDisplay"; },
    LBUF { txt = "ESecondaryKeyboard"; },
    LBUF { txt = "ESecondaryDisplayAndKeyboard"; },
    LBUF { txt = "ECustomTarget1"; },
    LBUF { txt = "ECustomTarget2"; },
    LBUF { txt = "ECustomTarget3"; },
    LBUF { txt = "ECustomTarget4"; }
    };
    }

RESOURCE DIALOG r_lightcontroller_lightstarget_list
    {
    flags = EGeneralQueryFlags;
    buttons = R_AVKON_SOFTKEYS_OK_CANCEL;
    items =
        {
        DLG_LINE
            {
            type = EAknCtListQueryControl;
            id = EListQueryControl;
            control = AVKON_LIST_QUERY_CONTROL
                {
                listtype = EAknCtSinglePopupMenuListBox;
                heading = "Lights target:";
                listbox = LISTBOX
                    {
                    flags = EAknListBoxSelectionList;
                    array_id = r_lightcontroller_lightstarget_items;
                    };
                };
            }
        };
    }

RESOURCE AVKON_LIST_QUERY r_lightcontroller_restore_list
    {
    flags = EGeneralQueryFlags;
    items =
        {
        AVKON_LIST_QUERY_DLG_LINE
            {
            type = EAknCtListQueryControl;
            id = EListQueryControl;
            control = AVKON_LIST_QUERY_CONTROL
                {
                listtype = EAknCtSinglePopupMenuListBox;
                heading = "Restore parameter:";
                listbox = LISTBOX
                    {
                    flags = EAknListBoxSelectionList;
                    array_id = r_lightcontroller_bool_items;
                    };
                };
            }
        };
    }

RESOURCE AVKON_LIST_QUERY r_lightcontroller_noccoeenv_list
    {
    flags = EGeneralQueryFlags;
    items =
        {
        AVKON_LIST_QUERY_DLG_LINE
            {
            type = EAknCtListQueryControl;
            id = EListQueryControl;
            control = AVKON_LIST_QUERY_CONTROL
                {
                listtype = EAknCtSinglePopupMenuListBox;
                heading = "NoCCoeEnv parameter:";
                listbox = LISTBOX
                    {
                    flags = EAknListBoxSelectionList;
                    array_id = r_lightcontroller_bool_items;
                    };
                };
            }
        };
    }

RESOURCE AVKON_LIST_QUERY r_lightcontroller_fade_list
    {
    flags = EGeneralQueryFlags;
    items =
        {
        AVKON_LIST_QUERY_DLG_LINE
            {
            type = EAknCtListQueryControl;
            id = EListQueryControl;
            control = AVKON_LIST_QUERY_CONTROL
                {
                listtype = EAknCtSinglePopupMenuListBox;
                heading = "Fade parameter:";
                listbox = LISTBOX
                    {
                    flags = EAknListBoxSelectionList;
                    array_id = r_lightcontroller_bool_items;
                    };
                };
            }
        };
    }
```

Header file

```
#ifndef LIGHTCONTROLLER_H
#define LIGHTCONTROLLER_H

#include <HWRMLight.h>
```


(C) Copyright Nokia 2013. All rights reserved.

```
class CLightController : public CBase,
                         public MHWRMLightObserver

    {
public:
    static CLightController*  NewL();
        ~CLightController();

        //ask reserve parameters with dialogs
        void LightsReserveL();
        void LightsReserveL(TInt aTarget,
                  TBool aRestoreState,
                  TBool aForceNoCCoeEnv);

        //ask light on parameters with dialogs
        void LightsOnL();
        void LightsOnL(TInt aTarget,
                    TInt aDuration,
                    TInt aIntensity,
                    TBool aFadeIn);

        //ask light blink parameters with dialogs
        void LightsBlinkL();
        void LightsBlinkL(TInt aTarget,
                     TInt aDuration,
                     TInt aOnDuration,
                     TInt aOffDuration,
                     TInt aIntensity);

        //ask light off parameters with dialogs
        void LightsOffL();
        void LightsOffL(TInt aTarget,
                   TInt aDuration,
                   TBool aFadeOut);
        void LightsReleaseL(TInt aTarget=0);

        CHWRMLight::TLightStatus LightsStatusL(TInt aTarget=0);
private:

        void ConstructL();

        // from MHWRMLightObserver
        virtual void LightStatusChanged(TInt aTarget,
                                        CHWRMLight::TLightStatus aStatus);

    // returns the target or KErrCancel if user pressed cancel
    TInt GetLightsTargetL();
private:
        CHWRMLight*                  iLight;
    };

#endif  // LIGHTCONTROLLER_H
```

Source file

```
#include <aknNoteWrappers.h> //CAknListQueryDialog,CAknNumberQueryDialog
#include <LightController.rsg>
#include "LightController.h"

CLightController* CLightController::NewL()
  {
  CLightController* self=
      new(ELeave) CLightController( );
  CleanupStack::PushL(self);
  self->ConstructL();
  CleanupStack::Pop(self);
  return self;
  }

void CLightController::ConstructL()
    {
    iLight = CHWRMLight::NewL(this);
    }

CLightController::~CLightController()
    {
    delete iLight;
    }

void CLightController::LightsReserveL(TInt aTarget,
                  TBool aRestoreState,
                  TBool aForceNoCCoeEnv)

    {
    if(aTarget <= CHWRMLight::ENoTarget || aTarget > CHWRMLight::ECustomTarget4)
      LightsReserveL();
    else
      iLight->ReserveLightL( aTarget, aRestoreState, aForceNoCCoeEnv );
    }

void CLightController::LightsReserveL()
    {
    TInt item(0);
    TBool restore;
    TBool forceNoCCoeEnv;
    TInt target(CHWRMLight::ENoTarget);

    // get the lights target
    target = GetLightsTargetL();

    if( target == KErrCancel )
        {
        // no target entered, just return
        return;
        }

    CAknListQueryDialog* listDlg = new (ELeave)CAknListQueryDialog( &item );

    if( listDlg->ExecuteLD(R_LIGHTCONTROLLER_RESTORE_LIST) )
        {
        // got restore state parameter
        item == 0 ? restore = EFalse : restore = ETrue;
        //now get force no CCoeEnv parameter
        listDlg = new (ELeave)CAknListQueryDialog( &item );
```


(C) Copyright Nokia 2013. All rights reserved.

```
        if( listDlg->ExecuteLD(R_LIGHTCONTROLLER_NOCCOEENV_LIST) )
            {
            // got ForceNoCCoeEnv parameter
            item == 0 ? forceNoCCoeEnv = EFalse : forceNoCCoeEnv = ETrue;
            iLight->ReserveLightL( target, restore, forceNoCCoeEnv );
            }
        else
            {
            // did not get ForceNoCCoeEnv parameter
            // do not call ReserveLightL() at all
            }
        }
    else
        {
        // did not get restore parameter
        iLight->ReserveLightL( target );
        }

    }

void CLightController::LightsReleaseL(TInt aTarget)
    {
    TInt target(CHWRMLight::ENoTarget);

    if(aTarget <= CHWRMLight::ENoTarget || aTarget > CHWRMLight::ECustomTarget4)
      {
      // get the lights target
      target = GetLightsTargetL();
      }
    else
      {
      target = aTarget;
      }

    if( target == KErrCancel )
        {
        // no target entered
        // do not call ReleaseLight at all
        }
    else
        {
        //call ReleaseLight with the given target
        iLight->ReleaseLight( target );
        }
    }

void CLightController::LightsOnL(TInt aTarget,
                    TInt aDuration,
                    TInt aIntensity,
                    TBool aFadeIn)
  {


   if(aTarget <= CHWRMLight::ENoTarget || aTarget > CHWRMLight::ECustomTarget4 ||
      aDuration < KHWRMInfiniteDuration || aDuration > KHWRMLightMaxDuration ||
      aIntensity < KHWRMDefaultIntensity || aIntensity > KHWRMLightMaxIntensity)
        {
        LightsOnL();
        }
   else
     {
     iLight->LightOnL(aTarget, aDuration,
      aIntensity, aFadeIn);
     }
  }

void CLightController::LightsOnL()
    {
    TInt item(0);
    TInt duration(0);
    TInt intensity(0);
    TInt target(CHWRMLight::ENoTarget);
    TBool fade(EFalse);

    // get the lights target
    target = GetLightsTargetL();

    if( target == KErrCancel )
        {
        // no target entered
        // do not call LightOnL at all
        return;
        }

    //duration
    CAknNumberQueryDialog* queryDlg = CAknNumberQueryDialog::NewL( duration );
    if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_DURATION_QUERY ) )
        {
        // got duration so get intensity
        queryDlg = CAknNumberQueryDialog::NewL( intensity );
        if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_LIGHT_INTENSITY_QUERY ) )
            {
            // got the intensity
            CAknListQueryDialog* listDlg = new (ELeave)CAknListQueryDialog( &item );
            if( listDlg->ExecuteLD(R_LIGHTCONTROLLER_FADE_LIST) )
                {
                // got fade parameter
                item == 0 ? fade = EFalse : fade = ETrue;
                iLight->LightOnL( target, duration, intensity, fade );
                }
            else
                {
                // did not get fade parameter
                // do not call LightOnL() at all
                }
            }
        else
            {
            // user pressed cancel for intensity query
            iLight->LightOnL( target, duration );
            }
        }
    else
        {
        // didn't get duration so call basic API
        iLight->LightOnL( target );
        }
    }

void CLightController::LightsBlinkL(TInt aTarget,
```


(C) Copyright Nokia 2013. All rights reserved.

```
                    TInt aDuration,
                    TInt aOnDuration,
                    TInt aOffDuration,
                    TInt aIntensity)
  {
   if(aTarget <= CHWRMLight::ENoTarget || aTarget > CHWRMLight::ECustomTarget4 ||
    aDuration < KHWRMInfiniteDuration || aDuration > KHWRMLightMaxDuration ||
    aOnDuration < KHWRMInfiniteDuration || aOnDuration > KHWRMLightMaxDuration ||
    aOffDuration < KHWRMInfiniteDuration || aOffDuration > KHWRMLightMaxDuration ||
    aIntensity < KHWRMDefaultIntensity || aIntensity > KHWRMLightMaxIntensity)
     {
     LightsBlinkL();
     }
   else
     {
     iLight->LightBlinkL(aTarget, aDuration, aOnDuration,
                aOffDuration, aIntensity);
     }
  }

void CLightController::LightsBlinkL()
    {
    TInt duration(0);
    TInt onDuration(0);
    TInt offDuration(0);
    TInt intensity(0);

    // get the lights target
    TInt target = GetLightsTargetL();
    if( target == KErrCancel )
        {
        // no target entered, just return
        return;
        }

    // get duration
    CAknNumberQueryDialog* queryDlg = CAknNumberQueryDialog::NewL( duration );
    if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_DURATION_QUERY ) )
        {
        // got duration so get OnDuration
        queryDlg = CAknNumberQueryDialog::NewL( onDuration );
        CleanupStack::PushL( queryDlg );

        _LIT(KEnterONDuration, "Enter ON duration (0 to 2147482)");
        queryDlg->SetPromptL( KEnterONDuration );

        CleanupStack::Pop( queryDlg );
        if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_DURATION_QUERY ) )
            {
            // got OnDuration so get OffDuration
            queryDlg = CAknNumberQueryDialog::NewL( offDuration );
            CleanupStack::PushL( queryDlg );

            _LIT(KEnterOFFDuration, "Enter OFF duration (0 to 2147482)");
            queryDlg->SetPromptL( KEnterOFFDuration );

            CleanupStack::Pop( queryDlg );
            if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_DURATION_QUERY ) )
                {
                // got OffDuration so get intensity
                queryDlg = CAknNumberQueryDialog::NewL( intensity );
                if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_LIGHT_INTENSITY_QUERY ) )
                    {
                    // got the intenstity
                    iLight->LightBlinkL( target, duration, onDuration,
                                         offDuration, intensity );
                    }
                else
                    {
                    // user pressed cancel for intensity
                    // do not call LightBlinkL() at all
                    }
                }
            else
                {
                // user pressed cancel for OffDuration
                // do not call LightBlinkL() at all
                }
            }
        else
            {
            // user pressed cancel for OnDuration
            iLight->LightBlinkL( target, duration );
            }
        }
    else
        {
        // user pressed cancel for Duration
        iLight->LightBlinkL( target );
        }
    }

void CLightController::LightsOffL(TInt aTarget,
                    TInt aDuration,
                    TBool aFadeOut)
  {
   if(aTarget <= CHWRMLight::ENoTarget || aTarget > CHWRMLight::ECustomTarget4 ||
       aDuration < KHWRMInfiniteDuration || aDuration > KHWRMLightMaxDuration)
     {
     LightsOffL();
     }
   else
     {
     iLight->LightOffL(aTarget, aDuration, aFadeOut);
     }
  }

void CLightController::LightsOffL()
    {
    TInt item(0);
    TInt duration(0);
    TInt target(CHWRMLight::ENoTarget);
    TBool fade(EFalse);

    // get the lights target
    target = GetLightsTargetL();

    if( target == KErrCancel )
        {
        // no target entered, just return
```


(C) Copyright Nokia 2013. All rights reserved.

```
        return;
        }

    //duration
    CAknNumberQueryDialog* queryDlg = CAknNumberQueryDialog::NewL( duration );
    if( queryDlg->ExecuteLD( R_LIGHTCONTROLLER_DURATION_QUERY ) )
        {
        // got duration so get fade
        CAknListQueryDialog* listDlg = new (ELeave)CAknListQueryDialog( &item );
        if( listDlg->ExecuteLD(R_LIGHTCONTROLLER_FADE_LIST) )
            {
            // got fade parameter
            item == 0 ? fade = EFalse : fade = ETrue;
            iLight->LightOffL( target, duration, fade );
            }
        else
            {
            // did not get fade parameter
            iLight->LightOffL( target, duration );
            }
        }
    else
        {
        // didn't get duration so call basic API
        iLight->LightOffL( target );
        }
    }

CHWRMLight::TLightStatus CLightController::LightsStatusL(TInt aTarget)
    {
    TInt target(CHWRMLight::ENoTarget);

    if(aTarget <= CHWRMLight::ENoTarget || aTarget > CHWRMLight::ECustomTarget4)
      {
      // get the lights target
      target = GetLightsTargetL();
      }
    else
      {
      target = aTarget;
      }

    if( target == KErrCancel )
        {
        // no target entered
        return CHWRMLight::ELightStatusUnknown;
        }
    else
        {
        return iLight->LightStatus( target );
        }
    }

TInt CLightController::GetLightsTargetL()
    {
    TInt item(0);
    TInt ret(KErrCancel);

    CAknListQueryDialog* listDlg = new (ELeave)CAknListQueryDialog( &item );
    if( listDlg->ExecuteLD(R_LIGHTCONTROLLER_LIGHTSTARGET_LIST) )
        {
        // This is a lookup table for the items defined in the
        // resource array r_lightcontroller_lightstarget_items
        switch ( item )
            {
            case 0: // ENoTarget
                {
                ret = CHWRMLight::ENoTarget;
                break;
                }
            case 1: // EPrimaryDisplay
                {
                ret = CHWRMLight::EPrimaryDisplay;
                break;
                }
            case 2: // EPrimaryKeyboard
                {
                ret = CHWRMLight::EPrimaryKeyboard;
                break;
                }
            case 3: // EPrimaryDisplayAndKeyboard
                {
                ret = CHWRMLight::EPrimaryDisplayAndKeyboard;
                break;
                }
            case 4: // ESecondaryDisplay
                {
                ret = CHWRMLight::ESecondaryDisplay;
                break;
                }
            case 5: // ESecondaryKeyboard
                {
                ret = CHWRMLight::ESecondaryKeyboard;
                break;
                }
            case 6: // ESecondaryDisplayAndKeyboard
                {
                ret = CHWRMLight::ESecondaryDisplayAndKeyboard;
                break;
                }
            case 7: // ECustomTarget1
                {
                ret = CHWRMLight::ECustomTarget1;
                break;
                }
            case 8: // ECustomTarget2
                {
                ret = CHWRMLight::ECustomTarget2;
                break;
                }
            case 9: // ECustomTarget3
                {
                ret = CHWRMLight::ECustomTarget3;
                break;
                }
            case 10: // ECustomTarget4
                {
                ret = CHWRMLight::ECustomTarget4;
                break;
                }
```


(C) Copyright Nokia 2013. All rights reserved.

```
            default:
                break;
            }
        }
    else
        {
        // no target entered
        ret = KErrCancel;
        }

    return ret;
    }

void CLightController::LightStatusChanged(TInt aTarget,
                      CHWRMLight::TLightStatus aStatus)
    {
      if(!aTarget)
        {
        //No target specified
        }

        switch ( aStatus )
            {
            case CHWRMLight::ELightOn:
                {
              //ELightOn do something...
              break;
                }
            case CHWRMLight::ELightOff:
                {
                //ELightOff do something...
                break;
                }
            case CHWRMLight::ELightBlink:
                {
                //ELightBlink do something...
                break;
                }
            case CHWRMLight::ELightStatusUnknown:
                {
                //ELightStatusUnknown do something...
                break;
                }
            default:
                {
                //Lights status undefined do something...
                break;
                }
            }

    }
```

Using CLightController class

- In the header file:

```
class CLightController;

class CTestingAppUi : public CAknAppUi
  {
  //...
private:
  //...
  CLightController* iLightController;
  };
```

- In the source file:

```
#include "LightController.h"
#include <HWRMLight.h>

void CTestingAppUi::ConstructL()
  {
  iLightController = CLightController::NewL();
  }

CTestingAppUi::~CTestingAppUi()
  {
  delete iLightController;
  }

void CTestingAppUi::ControlLightsL()
  {
  CHWRMLight::TLightTarget target = CHWRMLight::EPrimaryDisplayAndKeyboard;

  // == RESERVE ==
  /*
    1. target - the light device that should be controlled (display, keyboard,
    hardware-specific custom target)
    2. param - restore: if ETrue is selected, the light state on last release
    will be restored
    upon successful reservation.
    3. param - NoCCoeEnv: if EFalse is selected, the light target will be automatically
    released when the
    application goes to background and reserved when the application is restored
   to foreground.
  */

  //show dialogs to ask values
  iLightController->LightsReserveL();
  //...or set values without dialogs
  iLightController->LightsReserveL(target, EFalse, EFalse);

  // == LIGHTS ON ==
  /*
   1. param - target: the light device that should be controlled (display,
   keyboard, hardware specific custom target)
   2. param - duration: time in milliseconds light target will be active
   3. param - intensity: percentage of full light intensity
   4. param - fade: if ETrue is selected, the light target will be
   turned on/off gradually (fade-in/fade-out)
  */

  //show dialogs to ask values
```


(C) Copyright Nokia 2013. All rights reserved.

```
iLightController->LightsOnL();
//...or set values without dialogs
iLightController->LightsOnL(target, 2000, 50, EFalse);

// == LIGHTS BLINK ==
/*
 1. param - target: the light device that should be controlled (display,
 keyboard, hardware-specific custom target)
 2-4. - params durations: times in milliseconds that the light target will be active
 5. param - intensity: percentage of full light intensity
*/

//show dialogs to ask values
 iLightController->LightsBlinkL();
//...or set values without dialogs
iLightController->LightsBlinkL(target, 2000, 1000, 1000, 50);

// == LIGHTS OFF ==
/*
 1. param - target: the light device that should be controlled (display, keyboard,
 hardware-specific custom target)
 2. param - duration: time in milliseconds that the light target will be active
 3. param - fade: if ETrue is selected, the light target will be turned
 on/off gradually (fade-in/fade-out)
*/
//show dialogs to ask values
iLightController->LightsOffL();
//...or set values without dialogs
iLightController->LightsOffL(target, 2000, EFalse);

// == RELEASE ==
//show dialogs to ask target
iLightController->LightsReleaseL();
//...or set target without a dialog
iLightController->LightsReleaseL(target);

// == GET STATUSES
//show dialog to ask target
CHWRMLight::TLightStatus status(CHWRMLight::ELightStatusUnknown);

status = iLightController->LightsStatusL();
//...or set target without a dialog
status = iLightController->LightsStatusL(target);
}
```

Postconditions

The CLightController class controls light settings and receives information on light target status changes.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.