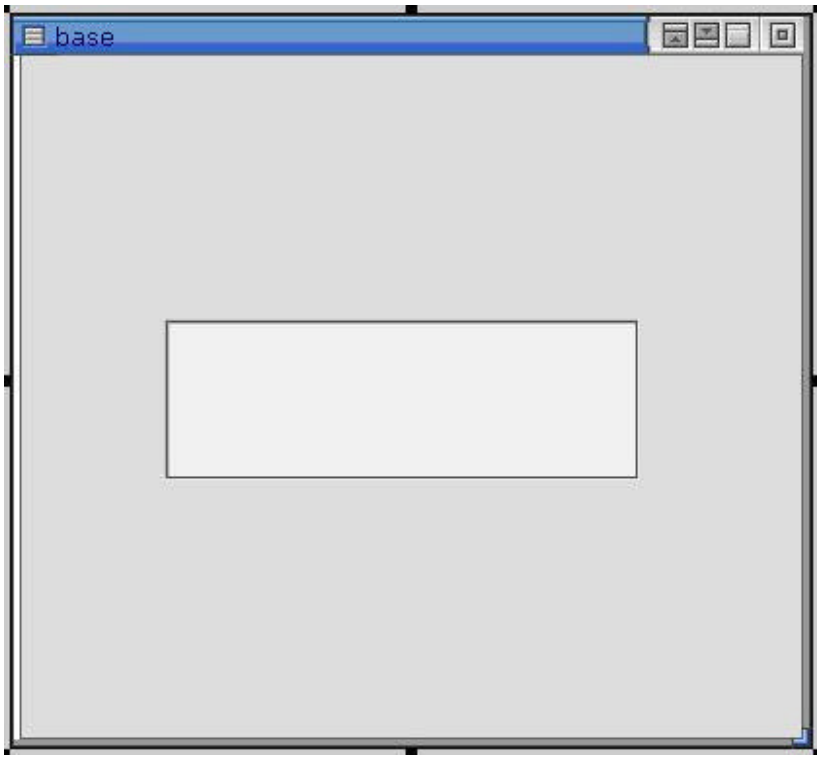
Multithreading und Photon

Es soll eine einfache Anzeige für die Uhrzeit das Datum entwickelt werden. In einem Textfenster sollen abwechselnd die Uhrzeit und das Datum angezeigt werden. Die Bereitstellung der Daten und das Zeitverhalten sollen in einem Thread realisiert werden.



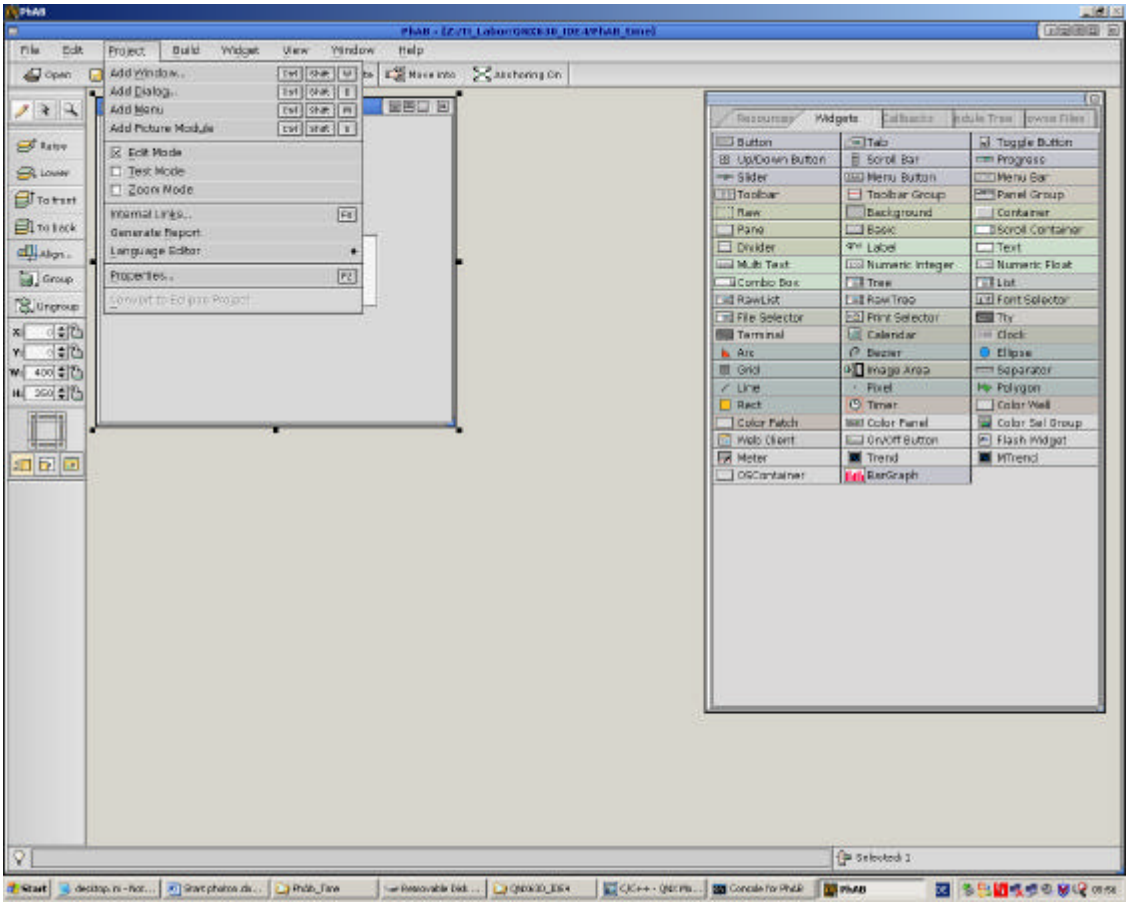
Photon ist, wie die meisten grafischen Oberflächen, nicht Multithreading fähig. Damit ein eigenständiger Thread ausgeführt werden kann muss

- während der Initialisierung der Anwendung eine eigene Methode ausgeführt werden, die diesen Thread startet.
- Der Zugriff auf die Resourcen eines Widgets mit den Zugriffen von Photon synchronisiert werden.

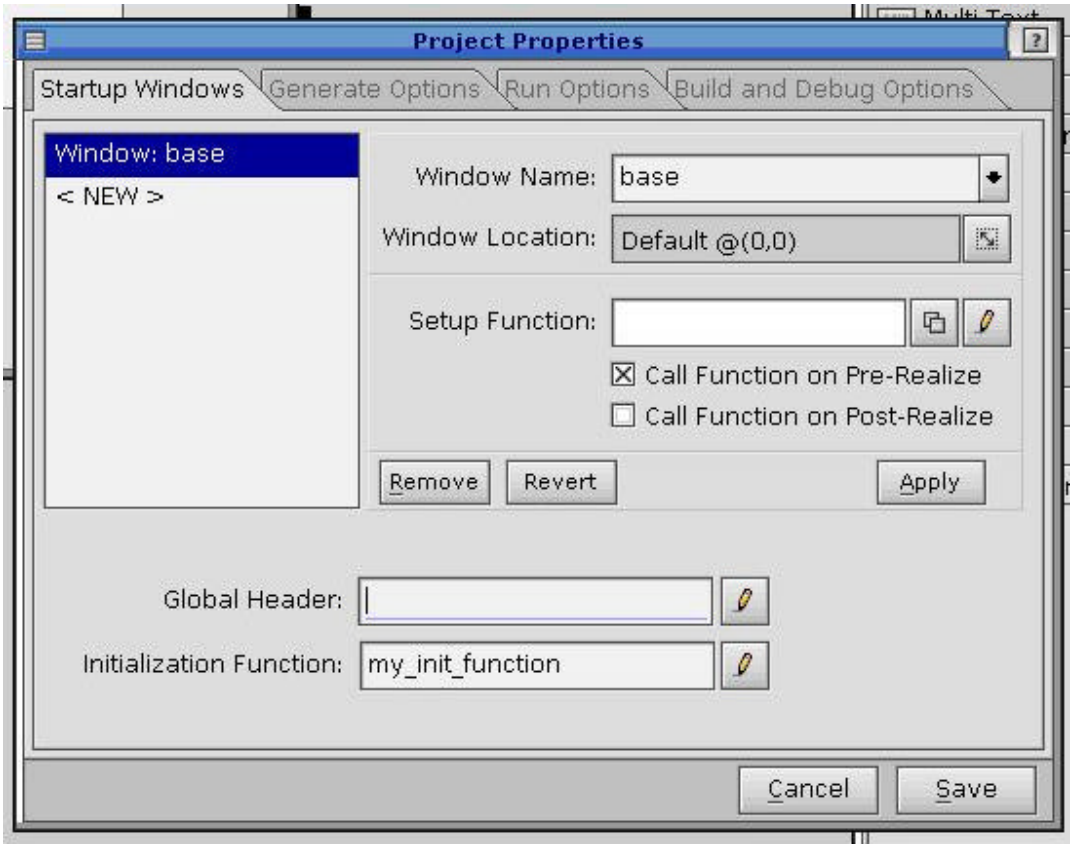
Eine eigne Initialisierungsmethode wird eingefügt in dem man unter

Project → Properties

anklickt.



In diesem Fenster wird unter “Initialization Function” der Name einer Methode eingetragen, die in diesem Fall den Code zum Starten eines Threads enthält.



Die Methode **my_init_function()** wird als Methodenrumpf vom Photon Application Builder zur Verfügung gestellt.

In dieser Methode wird im einfachsten Fall nur der Thread gestartet

```
/* Y o u r   D e s c r i p t i o n                       */
/*                            AppBuilder Photon Code Lib */
/*                                         Version 2.03  */

/* Standard headers */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>

/* Local headers */
#include "ablibs.h"
#include "abimport.h"
#include "proto.h"

#include <pthread.h>

void *time_date_thread(void *arg);

/* Application Options string */
const char ApOptions[] =
      AB_OPTIONS ""; /* Add your options in the "" */


int
my_init_function( int argc, char *argv[] )

      {

      /* eliminate 'unreferenced' warnings */
      argc = argc, argv = argv;

      pthread_create(NULL, NULL, &time_date_thread, NULL);

      return( Pt_CONTINUE );

      }
```

.

Der Zugriff auf die Resourcen des Widgets mittels **PtSetResource()** oder **PtGetResource()** muss mit den Zugriffen von Photon synchronisiert werden. Nach dem Aufruf von **PtEnter()** hat die Methode freien Zugriff. Photon kann erst wieder aktiv werden, nachdem die Methode PtLeave() ausgeführt worden ist.

```
#include <pthread.h>
#include <time.h>
#include <unistd.h>

/* Local headers */
#include "ablibs.h"
#include "abimport.h"
#include "proto.h"

void *time_date_thread(void *arg){

	int i;

	struct tm *zeit;
	time_t sekunde;
	char string[80];

	for(i = 0; i < 30; i++){

		sleep(3);
		time(&sekunde);
		zeit = localtime(&sekunde);
		strftime(string, 80, "%H : %M : %S", zeit);

		PtEnter(0);
		PtSetResource(ABW_PtText_date_time, Pt_ARG_TEXT_STRING,
				string, 0 );
		PtLeave(0);

		sleep(3);
		time(&sekunde);
		zeit = localtime(&sekunde);
		strftime(string, 80, "%d - %B - %Y", zeit);

		PtEnter(0);
		PtSetResource(ABW_PtText_date_time, Pt_ARG_TEXT_STRING,
				string, 0 );
		PtLeave(0);
	}


	return NULL;
}
```