[Q]: Rexx subcommand handler - пример

[A]: Dmitry Zavalishin (2:5020/32)

/ U1 *\ * * **Copyright (C) 1991-1995 by Infinity Soft * * Module : Rexx subcommand handler * * $Log: RexxScom.c $ * Revision 1.2 1995/11/05 13:52:48 dz * current. * * Revision 1.1 1995/05/08 16:04:26 dz * Initial revision * *** /

#define INCL_RXFUNC #define INCL_RXSUBCOM #define INCL_RXSHV #define INCL_REXXSAA #include <rexxsaa.h> /* needed for RexxStart() */ #include <stdio.h> /* needed for printf() */ #include <string.h> /* needed for strlen() */ #include <strng.h> /* needed for strlen() */

APIRET EXPENTRY U1_Command(PRXSTRING cmd, PUSHORT flags, PRXSTRING ret);

bool Register_Rexx_Subcommand_Handler(void)

```
{
```

```
RexxRegisterSubcomExe("U1", (PFN)U1_Command, NULL);
```

```
return Ok;
}
```

#define TEST(v, s) (strncmp(v, s, sizeof(s) - 1) == 0)

#define SC_SUCCESS { strcpy(ret→strptr, “0”); ret→strlength = 1; return 0; } #define SC_FAILURE(code) { *flags = RXSUBCOM_FAILURE; strcpy(ret→strptr, code); ret→strlength = 1; return 0; } #define SC_ERROR(code) { *flags = RXSUBCOM_ERROR; strcpy(ret→strptr, code); ret→strlength = 1; return 0; }

#define CMD(tail) { if((rc = sc_##tail(cmd)) != 0) { sprintf(rcs, “%d”, rc); SC_ERROR(rcs); } }

static sc_log(string &); static sc_warning(string &); static sc_error(string &); static sc_fatal(string &);

APIRET EXPENTRY U1_Command(PRXSTRING r_cmd, PUSHORT flags, PRXSTRING ret)

```
{
string cmd( r_cmd->strptr );
const char *p1, *p2;
const maxv = 25;
char  verb[maxv];
```

```
// for CMD macro
int rc;
char    rcs[10];
```

```
p1 = cmd.c_str();
p2 = strpbrk( p1, " \t" );
strncpy( verb, p1, min( maxv, p2-p1 ) );
verb[min( maxv, p2-p1 )] = '\0';
```

```
strlwr( verb );
```

```
while( *p2 == ' ' || *p2 == '\t' )
    p2++;
```

```
cmd = p2;
```

```
debug( "Rexx cmd got verb '%s' and tail '%s'", verb, cmd.c_str() );
```

```
if( TEST( verb, "log" ) ) CMD( log  )    else
if( TEST( verb, "warning" ) ) CMD( warning )    else
if( TEST( verb, "error" ) )   CMD( error  )    else
if( TEST( verb, "fatal" ) )   CMD( fatal  )    else
    {
    error( EI_None, "Rexx subcommand: unknown verb '%s'", verb );
    SC_FAILURE("33");
    }
```

```
SC_SUCCESS;
}
```

static int sc_log(string &s)

```
{
log( "x#", "%s", s.c_str() );
return 0;
}
```

static int sc_warning(string &s)

```
{
warning( EI_None, "%s", s.c_str() );
return 0;
}
```

static int sc_error(string &s)

```
{
error( EI_None, "%s", s.c_str() );
return 0;
}
```

static int sc_fatal(string &s)

```
{
fatal( EC_Dunno, EI_None, "%s", s.c_str() );
return 0;
}
```