Subject: Re: MT assertion failed in IsST()
Posted by mirek on Fri, 08 Jun 2007 07:29:55 GMT
View Forum Message <> Reply to Message

A *nasty* bug in U++ ("Format" function, called in httpcli, initialized formatters, which can be done in ST mode only).

Now fixed.

Quick patch:

Core/Format.cpp 808:

```
static void sRegisterFormatters()
{
 ONCELOCK {
  IntDoubleRegister(BOOL_V);
  IntDoubleRegister(INT_V);
  IntDoubleRegister(INT64_V);
  IntDoubleRegister(DOUBLE_V);

  RegisterStringFormatter("s", &StringFormatter);
  RegisterNullFormatter("", &DateFormatter);
  RegisterFormatter(DATE_V, "", &DateFormatter);
  RegisterFormatter(TIME_V, "", &TimeFormatter);

  RegisterNumberFormatter("n",  &RealFormatter);
  RegisterNumberFormatter("ne", &RealFormatter);
  RegisterNumberFormatter("nf", &RealFormatter);
  RegisterNumberFormatter("nl", &RealFormatter);
  RegisterNumberFormatter("nle", &RealFormatter);
  RegisterNumberFormatter("nlf", &RealFormatter);
  RegisterNumberFormatter("v",  &RealFormatter);
  RegisterNumberFormatter("ve", &RealFormatter);
  RegisterNumberFormatter("vf", &RealFormatter);
  RegisterNumberFormatter("vl", &RealFormatter);
  RegisterNumberFormatter("vle", &RealFormatter);
  RegisterNumberFormatter("vlf", &RealFormatter);

  // real number formats (n = fixed decimals, v = valid decimals)
  // ne, ve - force exponential notation; nf, vf - force fixed notation; nl, vl - language-based
formatting
  // Options: [+][[-]<digits>][!][^[+]<expdig>]
  // + .. always prepend sign
  // [-]<digits> .. number of decimals to print (negative = left of decimal point, default = 6)
  // ! .. keep insignificant zeros
  // ^ .. exponent options:
```

```
  // + .. always prepend sign to exponent
  // <expdig> exponent padding width

  RegisterNumberFormatter("a", &IntLowerAlphaFormatter);
  RegisterNumberFormatter("A", &IntUpperAlphaFormatter);
  RegisterNumberFormatter("r", &IntLowerRomanFormatter);
  RegisterNumberFormatter("R", &IntUpperRomanFormatter);

  RegisterValueFormatter("vt", &StdFormatFormatter);
  RegisterValueFormatter("", &StdFormatFormatter);
 }
}

INITBLOCK {
 sRegisterFormatters();
}

String NFormat(int language, const char *s, const Vector<Value>& v)
{
 sRegisterFormatters();
 Formatting f;
 f.language = language;
 String result;
 int pos = 0;
 const char *b;
```

Mirek