Subject: Re: Squirrel - the programming language
Posted by Sender Ghost on Fri, 25 Nov 2011 07:55:07 GMT
View Forum Message <> Reply to Message

mdelfede wrote on Fri, 25 November 2011 08:42
BTW, looking into your example, it seems to me that the squirrel time is the whole of compiling+running the script (why running ?) and evaluating the function; it would be more interesting to take evaluation part of pre-compiled script separate.

Because without running, it will not find Squirrel function to get interpreted result.
Results for pre-compiled code:

```
1 / (1 - x * y + x - y) = -0.01851851852
fn->Execute() = -0.01851851852
sum = 5190404.858
sum = 5190404.858
sum = 5190404.858
sum = 5190404.858
TIMING Squirrel (fully interpreted): 195.00 ms - 195.00 ms (195.00 ms / 1 ), min: 195.00 ms, max: 195.00 ms, nesting: 1 - 1
TIMING Direct        : 13.00 ms - 13.00 ms (13.00 ms / 1 ), min: 13.00 ms, max: 13.00 ms, nesting: 1 - 1
TIMING Compiled      : 58.00 ms - 58.00 ms (58.00 ms / 1 ), min: 58.00 ms, max: 58.00 ms, nesting: 1 - 1
TIMING Interpreted   : 759.00 ms - 759.00 ms (759.00 ms / 1 ), min: 759.00 ms, max: 759.00 ms, nesting: 1 - 1
```

Results by sections:
Toggle Spoiler

```
using namespace Sqrat;

double sum = 0;
const String text =
"function compute() {\n"
" local x, y, sum = 0.0;\n"
" for (x = 0.0; x < 1; x += 0.001)\n"
"  for (y = 0.0; y < 1; y += 0.001)\n"
"   sum += 1 / (1 - x * y + x - y);\n"
" return sum;\n"
"}\n";

// creates a VM with initial stack size 1024
HSQUIRRELVM vm = sq_open(1024);

DefaultVM::Set(vm);

Script script;
```

```
{
 RTIMING("Squirrel (compiling)")
 try {
  script.CompileString(~text);
 }
 catch (Error ex) {
  Cerr() << "Exception: " << ex.Message(vm).c_str() << '\n';
  SetExitCode(1);
  return;
 }
}

Function compute;

{
 RTIMING("Squirrel (running and getting function)")
 try {
  script.Run();
 }
 catch(Error ex) {
  Cerr() << "Exception: " << ex.Message(vm).c_str() << '\n';
  SetExitCode(1);
  return;
 }

 compute = RootTable().GetFunction("compute");
}

{
 RTIMING("Squirrel (fully interpreted)")
 if (!compute.IsNull()) {
  try {
   SharedPtr<double> val = compute.Evaluate<double>();
   sum = *val;
  }
  catch (Error ex) {
   Cout() << "Exception: " << ex.Message(vm).c_str() << '\n';
   SetExitCode(1);
   return;
  }
 }
}

//sq_close(vm);
RDUMP(sum);
```

```
1 / (1 - x * y + x - y) = -0.01851851852
fn->Execute() = -0.01851851852
sum = 5190404.858
sum = 5190404.858
sum = 5190404.858
sum = 5190404.858
TIMING Squirrel (fully interpreted): 222.00 ms - 222.00 ms (222.00 ms / 1 ), min: 222.00 ms, max: 222.00 ms, nesting: 1 - 1
TIMING Squirrel (running and getting function):  0.00 ns -  0.00 ns ( 0.00 ns / 1 ), min:  0.00 ns, max:  0.00 ns, nesting: 1 - 1
TIMING Squirrel (compiling):  0.00 ns -  0.00 ns ( 0.00 ns / 1 ), min:  0.00 ns, max:  0.00 ns, nesting: 1 - 1
TIMING Direct         : 14.00 ms - 14.00 ms (14.00 ms / 1 ), min: 14.00 ms, max: 14.00 ms, nesting: 1 - 1
TIMING Compiled       : 59.00 ms - 59.00 ms (59.00 ms / 1 ), min: 59.00 ms, max: 59.00 ms, nesting: 1 - 1
TIMING Interpreted    : 848.00 ms - 848.00 ms (848.00 ms / 1 ), min: 848.00 ms, max: 848.00 ms, nesting: 1 - 1
```

Edit: Updated to Squirrel 3.0.7 version. The results from previous versions.

File Attachments

1) UppCompiler_with_Sqrat_2.zip, downloaded 350 times