Subject: Re: JavaScriptCore
Posted by dolik.rce on Sat, 10 Jul 2010 17:29:06 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 10 July 2010 16:23Sounds very very good.

Webkit itself, any chances? Anybody?

Hi Mirek,

I was hoping this will drag someones attention so I don't have to do the WebKit port all by myself But in worst case, I will definitely at least try...

I cleaned the things up a bit, so here are the "How to do this at home" instructions

Get the WebKit sources from SVN:
cd C:\upp
svn checkout http://svn.webkit.org/repository/webkit/trunk/JavaScriptCore WebKit\JavaScriptCoreYou could also get the full trunk, but for now it is not necessary
Patch the sources:
Download the attachment of this message, inside is directory JavaScriptCore, which contains .upp file and three files that need to be patched. The directory structure is the same, so it should be save just to copy it over the original directory tree.
Copy API headers:
Because of the way how WebKit is normally built, it expects some of the headers in JavaScriptCore\JavaScriptCore, so we have to copy it in there.
cd C:\upp\WebKit\JavaScriptCore
copy API*.h JavaScriptCore
Create look-up tables:
Some parts of JavaScript core use look-up tables which are usually generated in makefile, but we have to do it ourselves for now.
python create_regex_tables > yarr/RegExpTables.h
perl create_hash_table parser/Keywords.table > parser/Lexer.lut.h
perl create_hash_table runtime/RegExpConstructor.cpp -i > runtime/RegExpConstructor.lut.h
perl create_hash_table runtime/MathObject.cpp -i > runtime/MathObject.lut.h
perl create_hash_table runtime/JSONObject.cpp -i > runtime/JSONObject.lut.h
perl create_hash_table runtime/DatePrototype.cpp -i > runtime/DatePrototype.lut.h
perl create_hash_table runtime/ArrayPrototype.cpp -i > runtime/ArrayPrototype.lut.h
perl create_hash_table runtime/StringPrototype.cpp -i > runtime/StringPrototype.lut.h
perl create_hash_table runtime/RegExpObject.cpp -i > runtime/RegExpObject.lut.h
perl create_hash_table runtime/NumberConstructor.cpp -i > runtime/NumberConstructor.lut.h
perl pcre/dftables pcre/chartables.c Sorry for the slashes, I did this in cygwin so they are unix style, if you have windows perl or python, you have to rewrite them. In case you don't have perl or python I added them into the attached file as well. They are in the lut directory, just copy them to correct places.
Download ICU:
Download and extract the library from

http://download.icu-project.org/files/icu4c/4.4.1/icu4c-4_4_ 1-Win32-msvc9.zip. If you have 64b windows, you might need to download different file. I extracted this file into C:\upp\Webkit. If you put it somewhere else, you will have to change the following paths accordingly. Now it is necessary to make mingw aware of ICU, so we add C:\upp\WebKit\icu\include to include directories and C:\upp\WebKit\icu\lib in lib directories in build methods.
Setup assembly:
Create new assembly from the C:\upp\WebKit and C:\upp\uppsrc nests.
Get jsc package:
The last directory in the attachment file is jsc. That is the interpreter package. Just copy it into C:\upp\WebKit.

Doesn't look really difficult, right? The worst part was getting all the switches in wtf/Platform.h right
If I didn't forget something, you should be now able to compile, link and run jsc. If I did forget something, let me know

Honza

EDIT: Corrected paths in upp file and reuploaded the attachment.
EDIT2: I knew I forgot something One more lookup table, now added.
EDIT3: Hopefully last one... Forgot to mention ICU dlls (from C:\upp\WebKit\icu\bin) must be either on PATH or in the same directory as the final executable. But you all probably know that

File Attachments

1) jsc_all.zip, downloaded 326 times