RubyScript2Exe

Table of Contents

RubyScript2Exe

A Ruby Compiler

27 December 2004 13:45:10 CET

Erik Veenstra <rubyscript2exe@erikveen.dds.nl>

1. Introduction

RubyScript2Exe transforms your Ruby script into a standalone Windows or Linux executable. You can look at it as a "compiler". Not in the sense of a source−code−to−byte−code compiler, but as a "collector", for it collects all necessary files to run your script on an other machine: the Ruby script, the Ruby interpreter and the Ruby runtime library (stripped down for this script). Anyway, the result is the same: a standalone `exe`−file (Windows) or `bin`−file (Linux). And that's what we want!

Because of the gathering of files from your own Ruby installation, RubyScript2Exe creates an executable for the platform it's being run on. No cross compile.

And when I say Windows, I mean both Windows (RubyInstaller, MinGW and MSWin32) and Cygwin. But the generated `exe` under Cygwin is very, very big, because its `exe`'s are very big (static?) and it includes `cygwin1.dll`, so it can run on machines without Cygwin.

There is one more advantage: Because there might be some incompatibilities between the different Ruby versions, you have to test your script with every single version. Unless you distribute your version of Ruby with your script...

The combination of Tar2RubyScript and RubyScript2Exe is of special interest: A complete Ruby application can be distributed as one executable:

- Tar2RubyScript creates a standalone Ruby script (or RBA, Ruby Archive) of the application and its directory. This RBA can run on "the Ruby platform". This means that Ruby itself must be installed on the targeted system.

- RubyScript2Exe avoids this dependency by compiling a rubyscript (in casu the RBA), the Ruby interpreter and the Ruby runtime environment into one executable.

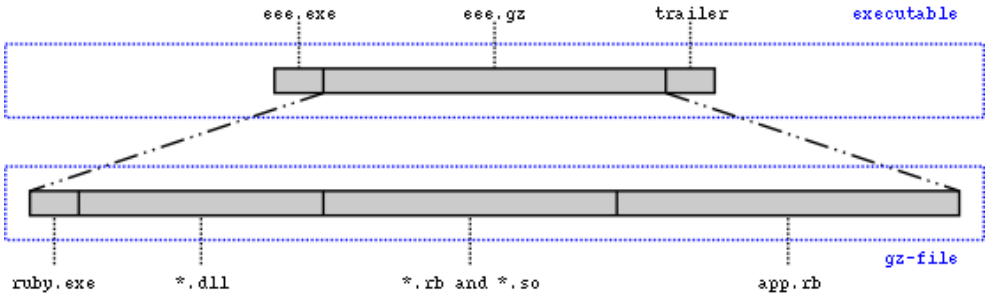
This combination isn't necessary. Each application can be used without the other.

What's the difference between RubyScript2Exe and AllInOneRuby? Well, RubyScript2Exe includes an application (your script) and only parts of the rubylib tree (it's stripped specifically for that application). AllInOneRuby contains a complete Ruby installation: it includes no application, but it does include the complete rubylib tree. You can use `allinoneruby.exe` like `ruby.exe` (Windows) and `allinoneruby.bin` like `ruby` (Linux) that's already installed on your system. In other words: the executable, generated with RubyScript2Exe, is an application; the one generated with AllInOneRuby "is" Ruby.

2. Internals

2.1. RubyScript2Exe

RubyScript2Exe monitors the execution of your script. After your script has finished, it gathers all program files and requirements (`ruby.exe`, `rubyw.exe` and `ruby` (and their `dll`'s and `so`'s, determined recursively), `*.rb` and `*.so` (and their `dll`'s and `so`'s, determined recursively)) from your own Ruby installation. All these files, your script and an extracting program are combined into one single, compressed executable. This executable can run on a bare Windows installation or a Linux installation with a libc version >= yours. Call it a "just−in−time and temporary installation of Ruby"...



The RubyInstaller (at least `ruby181−13.exe`) uses `msvcr71.dll`. This `dll` is distributed with your application, so the result might be bigger than the ones created with MinGW or MSWin32.

2.2. EEE

EEE stands for "Environment Embedding Executable". Well, I just had to give it a name...

EEE is the little Pascal program that combines and compresses all necessary files. It had to be written in a language that could be compiled and linked into an `exe`−file. Ruby wasn't an option. I use FreePascal, version 1.9.4.

EEE has two modes: packing and unpacking. When it detects an attached archive, it jumps into unpacking mode; into packing mode otherwise.

After creating the temporary directory and unpacking all files, EEE spawns the Ruby interpreter for your `app.rb` . At that point, EEE releases control to Ruby itself. After Ruby has finished, EEE regains control and starts cleaning up.

The difference between `eee.exe` and `eeew.exe` is the same as the difference between `ruby.exe` and `rubyw.exe`.

I've given EEE a page of its own, with more information then this section provides.

3. Usage

3.1. RubyScript2Exe

```
c:\home\erik> ruby rubyscript2exe.rb application[.rb[w]] [--rubyscript2exe-ruby|--rubyscript2exe-
```

If neither "rb" nor "rbw" is provided as extension, "rb" will be chosen as the default. So, "application.txt" will become "application.txt.rb" ...

If the extension is "rb", a DOS box will pop up. If the extension is "rbw", no DOS box will pop up. Unless it is overwritten by a parameter.

The option `--rubyscript2exe-rubyw` avoids the popping up of a DOS box. (It's annoying in the test period... No `puts` and `p` anymore... Only use it for distributing your application.) The option `--rubyscript2exe-ruby` forces the popping up of a DOS box (default).

On Linux, there's no difference between `ruby` and `rubyw`.

It is possible to change the icon of the generated executable manually, with a resource editor like Resource Hacker. If Resource Hacker is installed, in your %PATH% and therefor available from the current directory, and an icon file with the name *`application`*`.ico` exists in the current directory, the default icon will automatically be replaced by yours. I used Resource Hacker 3.4.0 for my tests.

3.2. The Executable (EEE)

To run the application:

```
c:\home\erik> application.exe
```

If you just want to list the contents of the executable, use: (Doesn't work in combination with `rubyw`.)

```
c:\home\erik> application.exe --eee-list
```

If you just want to extract the original files from the executable into the current directory, use:

```
c:\home\erik> application.exe --eee-justextract
```

3.3. Hacking on Location

You can extract, modify and "compile" on location, if you want to.

First, extract the executable:

```
c:\home\erik> application.exe --eee-justextract
```

This creates the directory `bin` (with the files `ruby.exe`, `rubyw.exe` and `*.dll`), the directory `lib` (with the dependencies `*.rb` and `*.so`), the directory `app` (with the file `app.rb`, which is your script) and the files `app.eee` and `eee.exe` (or `eeew.exe`) in the current directory.

It's possible to run your application again with:

```
c:\home\erik> bin\ruby.exe -I lib\ app\app.rb
```

If the application does a `Dir.chdir`, you have to do:

```
c:\home\erik> bin\ruby.exe -I ?:\full\path\to\lib\ app\app.rb
```

You can "compile" your application again with:

```
c:\home\erik> eee.exe app.eee newapplication.exe
```

... or ...

```
c:\home\erik> eeew.exe app.eee newapplication.exe
```

4. Examples

4.1. RubyScript2Exe

```
c:\home\erik> ruby rubyscript2exe.rb application.rb
```

This generates *application*.exe on Windows.

```
$ ruby rubyscript2exe.rb application.rb
```

This generates *application*.bin on Linux.

4.2. Distributions

On Windows, I ran RubyScript2Exe with four different Ruby distributions (Ruby 1.8.1):

Distribution	Size (bytes)
Cygwin	1287227
RubyInstaller	641840
MinGW	428898
MSWin32	467110

The details can be found here.

The test script was nothing more than a little Hello World thing (And the `require "rbconfig"` was just an extra test item...):

```
require "rbconfig"

puts "Hello World!"
```

4.3. Combination of Tar2RubyScript and RubyScript2Exe

Create a directory *application/* which contains at least `init.rb`.

```
c:\home\erik> ruby tar2rubyscript.rb application/
c:\home\erik> ruby rubyscript2exe.rb application.rb
```

This packs everything in *application/* into one executable. Ruby, the application and the directory it has to live in.

5. License

5.1. License of RubyScript2Exe

RubyScript2Exe, Copyright (C) 2003 Erik Veenstra <rubyscript2exe@erikveen.dds.nl>

This program is free software; you can redistribute it and/or modify it under the terms of the *GNU General Public License, version 2*, as published by the Free Software Foundation.

This program is distributed in the hope that it will be useful, **but without any warranty**; without even the implied warranty of **merchantability** or **fitness for a particular purpose**. See the *GNU General Public License* for more details.

You should have received a copy of the *GNU General Public License* along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place − Suite 330, Boston, MA 02111−1307, USA.

The full text of the license can be found here.

5.2. License of your Application

Whatever...

6. Download

Current version is 0.3.0 (27.12.2004). It's a stable release.

Tested on:

- Red Hat Linux 8.0 with Ruby 1.6.7
- Red Hat Linux 8.0 with Ruby 1.8.1
- Red Hat Linux 8.0 with Ruby 1.8.2
- Windows 95 with Ruby 1.8
- Windows 98 with Ruby 1.6 (Very slow!)
- Windows 98 with Ruby 1.8
- Windows 2000 with Ruby 1.8
- Windows 2000 with Ruby 1.8 (Cygwin)
- Windows XP with Ruby 1.8
- Windows XP with Ruby 1.8 (Cygwin)

You only need rubyscript2exe.rb . It's the current version, packed as an RBA (Ruby Archive, built by Tar2RubyScript) and works on both Windows and Linux. You can download rubyscript2exe.tar.gz if you want to play with the internals of RubyScript2Exe.

Send me reports of all bugs and glitches you find. Propositions for enhancements are welcome, too. This helps *us* to make *our* software better.

A change log and older versions can be found here. A generated log file can be found here.

RubyScript2Exe is available on SourceForge.net and on RubyForge .