Installing Nokogiri

Installing Nokogiri

Nokogiri includes its own updated and patched copies of libxml2 and libxslt libraries. By default, installation of Nokogiri will use these copies. Alternatively, you may install using your operating system's built-in libraries or other custom versions of these libraries.

This document also helps with common problems we've heard. If you have an issue not discussed here, please open an issue.

For earlier versions of Nokogiri, find instructions in the git history for this document.

For other platforms, please send pull requests to sparklemotion/nokogiri.org-tutorials.

Install with included libraries (RECOMMENDED)

Ubuntu / Debian

Install Nokogiri on a brand new Ubuntu system with these commands:

```
1 sudo apt-get install build-essential patch ruby-dev zlib1g-dev liblzma-dev
2 gem install nokogiri
```

Note for RVM users: you may require libgmp, consider running `sudo apt-get install libgmp-dev`.

FreeBSD / OpenBSD >= 6.2

Install Nokogiri on a brand new FreeBSD system with these commands:

```
1 gem install nokogiri
```

OpenBSD < 6.2

Use `gcc` from ports in order to compile the included libraries:

```
pkg_add -v gcc
gem install nokogiri
```

Windows

Use Rubyinstaller to install our binary distribution in a flash!

```
gem install nokogiri
```

Or build using DevKit. This is an advanced option you should only try if you know what you're doing. You MUST use Rubygems 2.4.5 or later.

Also see more details on this advanced procedure on this nokogiri-talk thread.

Red Hat / CentOS

Install Nokogiri on a brand new Red Had / CentOS system with these commands:

```
sudo yum install -y gcc ruby-devel zlib-devel
# sudo dnf install -y rpm-build # This may be required
gem install nokogiri
```

Alternatively, you may install the appropriate epel-release and get the Nokogiri package from EPEL using:

```
sudo yum install -y rubygem-nokogiri
```

GNU Guix

Install on any Linux distribution using GNU Guix, a reproducible binary software package management and distribution system.

Use this command:

```
guix package -i ruby-nokogiri
```

Note: source code is available here. A short description of how Nokogiri was packaged can be found here.

macOS

First, make sure you have the latest version of RubyGems and xcode commandline tools:

```
gem update --system
xcode-select --install # Then agree to the terms, even if you have done this before!
```

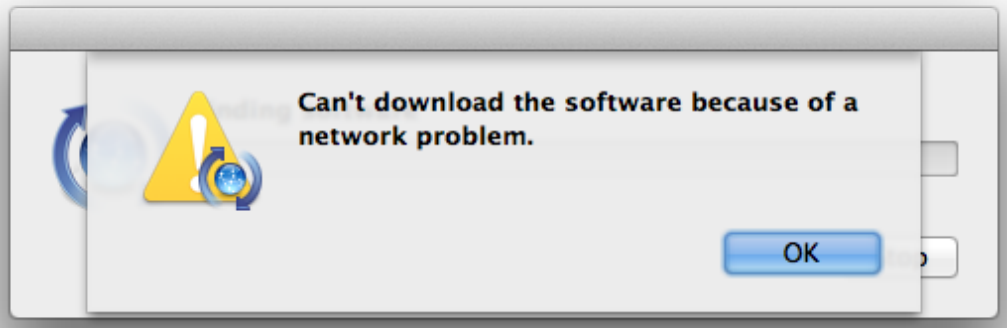
Then install nokogiri:

```
gem install nokogiri
```

This is verified working on maxOS 10.9 w/ Xcode's clang compiler. (Many thanks to @allaire and others for helping!)

`xcode-select` errors with a 'network problem'

If, you see this dialog when you run the above commands:



Then run this command to turn off forced-authentication with Apple Software Update:

```
sudo defaults delete /Library/Preferences/com.apple.SoftwareUpdate CatalogURL
```

Error Message About Undeclared Identifier `LZMA_OK`

A more recent error mentions an undeclared identifier `LZMA_OK`:

```
xmlIO.c:1450:52: error: use of undeclared identifier 'LZMA_OK'
    ret =  (__libxml2_xzclose((xzFile) context) == LZMA_OK ) ? 0 : -1;
                                                   ^

1 error generated.
```

The solution for this is a little more subtle and can be fixed in a couple of ways.

1. When using Homebrew, there are several libraries that use a formula called `xz` (including `the_silver_searcher` and `imagemagick`), which by default install a version of `liblzma` that is incompatible with most Ruby builds. (Homebrew installs only the 64-bit version of the library, but most Ruby builds are universal.) This can be fixed in a couple of ways:

    a. The most reliable way appears to be temporarily unlinking `xz` and relinking it during an install of `nokogiri`:

    ```
    brew unlink xz
    gem install nokogiri # or bundle install
    brew link xz
    ```

    b. The third way is to use a Homebrew-installed `libxml2`, as suggested in using your system libraries.

```
brew install libxml2
# If installing directly
gem install nokogiri -- --use-system-libraries \
  --with-xml2-include=$(brew --prefix libxml2)/include/libxml2
# If using Bundle
bundle config build.nokogiri --use-system-libraries \
  --with-xml2-include=$(brew --prefix libxml2)/include/libxml2
bundle install
```

When working with this, be certain to use `$(brew --prefix libxml2)` because it will use the correct location for your Homebrew install.

Other macOS Tips

- Make sure ruby is compiled with the latest clang compiler.
- Binary gems and ruby should be compiled with the same compiler/environment.
- If you have multiple versions of Xcode installed, make sure you use the right `xcode-select`.
- Try installing with system libraries.

If reporting an issue about the macOS installation instructions, please mention @zenspider.

Install with system libraries

Nokogiri will refuse to build against certain versions of `libxml2`, `libxslt` supplied with your operating system, and certain versions will cause mysterious problems. The compile scripts will warn you if you try to do this.

Step 1: Install `pkg-config`

On Debian/Ubuntu:

```
1 sudo apt-get install pkg-config
```

On FreeBSD:

```
1 sudo pkg install pkgconf
```

Step 2: Build Nokogiri

Using gem:

```
1 gem install nokogiri -- --use-system-libraries
```

Or, use Bundler:

```
1 bundle config build.nokogiri --use-system-libraries
2 bundle install
```

Install with custom / non-standard libraries

If:

- you've got `libxml2` and/or `libxslt` installed in a nonstandard place,
- and you don't have `pkg-config` installed

you can use command-line parameters to the `gem install` command to specify build parameters.

If you've got the proper `config` scripts:

```
gem install nokogiri -- \
    --use-system-libraries \
    --with-xml2-config=/path/to/xml2-config \
    --with-xslt-config=/path/to/xslt-config
```

or, you can specify the installation root directory:

```
gem install nokogiri -- \
    --use-system-libraries \
    --with-xml2-dir=/path/to/dir \
    --with-xslt-dir=/path/to/dir
```

or, you can specify include and library directories separately:

```
gem install nokogiri -- \
    --use-system-libraries \
    --with-xml2-lib=/path/to/builds/lib \
    --with-xml2-include=/path/to/builds/include/libxml2 \
    --with-xslt-lib=/path/to/builds/lib \
    --with-xslt-include=/path/to/builds/include
```

Note: By default, libxslt header files are installed into the root include directory, but libxml2 header files are installed into a subdirectory thereof named `libxml2`.

It's likely that you'll also need to specify the location of your `zlib` and `iconv` (and possibly `exslt`) install directories as well. In that case, you can add the options:

```
gem install nokogiri -- \
    --use-system-libraries \
    # ...
    --with-iconv-dir=/path/to/dir \
    --with-zlib-dir=/path/to/dir \
    [--with-exslt-dir=/path/to/dir]
    [--with-exslt-config=/path/to/exslt-config]
```

How to tell Bundler to use custom parameters

```
bundle config build.nokogiri \
       --use-system-libraries \
       --with-xml2-lib=/usr/local/lib \
       --with-xml2-include=/usr/local/include/libxml2/libxml \
       --with-xslt-lib=/usr/local/lib \
       --with-xslt-include=/usr/local/include/libxslt \
       --with-iconv-lib=/usr/local/lib \
       --with-iconv-include=/usr/local/include
```

Do not attempt Bundler installation using Bundler versions before v1.8.3. See related issue here. But if you really want to, see earlier git history of this file, which includes a workaround.

SmartOS (Nonstandard)

SmartOS installation requires building and using libxml2/libxslt/libiconv in a nonstandard location. Building on the previous section, here's how to do it:

(Note: `pkgsrc` is included in JPC SmartOS instances)

```
pkgin install ruby gcc49 libxml2 libxslt zlib libiconv ruby22-rake gmake
ln -s /opt/local/gcc49/bin/gcc /opt/local/bin/gcc

gem install nokogiri -- \
    --use-system-libraries \
    --with-xml2-lib=/opt/local/lib \
    --with-xml2-include=/opt/local/include/libxml2 \
    --with-xslt-lib=/opt/local/lib \
    --with-xslt-include=/opt/local/include/libxslt \
    --with-iconv-lib=/opt/local/lib \
    --with-iconv-include=/opt/local/include \
    --with-zlib-dir=/opt/local/lib
```

See the previous section for guidance on how to instruct Bundler to use these options.

Parsing an HTML/XML document →