Google Closure Compiler

build unknown

The Closure Compiler is a tool for making JavaScript download and run faster. It is a true compiler for JavaScript. Instead of compiling from a source language to machine code, it compiles from JavaScript to better JavaScript. It parses your JavaScript, analyzes it, removes dead code and rewrites and minimizes what's left. It also checks syntax, variable references, and types, and warns about common JavaScript pitfalls.

Getting Started

- Download the latest version
- See the Google Developers Site for documentation including instructions for running the compiler from the command line.

Options for Getting Help

1. Post in the Closure Compiler Discuss Group
2. Ask a question on Stack Overflow
3. Consult the FAQ

Building it Yourself

Note: The Closure Compiler requires Java 7 or higher.

Using Ant

1. Download the Ant build tool.

2. At the root of the source tree, there is an Ant file named `build.xml`. To use it, navigate to the same directory and type the command

   ```
   ant jar
   ```

   This will produce a jar file called `build/compiler.jar`.

Using Eclipse

1. Download and open the Eclipse IDE.
2. Navigate to `File > New > Project ...` and create a Java Project. Give the project a name.
3. Select `Create project from existing source` and choose the root of the checked-out source tree as the existing directory.
4. Navigate to the `build.xml` file. You will see all the build rules in the Outline pane. Run the `jar` rule to build the compiler in `build/compiler.jar`.

Running

On the command line, at the root of this project, type

```
java -jar build/compiler.jar
```

This starts the compiler in interactive mode. Type

```
var x = 17 + 25;
```

then hit "Enter", then hit "Ctrl-Z" (on Windows) or "Ctrl-D" (on Mac or Linux) and "Enter" again. The Compiler will respond:

```
var x=42;
```

The Closure Compiler has many options for reading input from a file, writing output to a file, checking your code, and running optimizations. To learn more, type

```
java -jar compiler.jar --help
```

More detailed information about running the Closure Compiler is available in the documentation.

Compiling Multiple Scripts

If you have multiple scripts, you should compile them all together with one compile command.

```
java -jar compiler.jar --js_output_file=out.js in1.js in2.js in3.js ...
```

You can also use minimatch-style globs.

```
# Recursively include all js files in subdirs
java -jar compiler.jar --js_output_file=out.js 'src/**.js'

# Recursively include all js files in subdirs, exclusing test files.
# Use single-quotes, so that bash doesn't try to expand the '!'
java -jar compiler.jar --js_output_file=out.js 'src/**.js' '!**_test.js'
```

The Closure Compiler will concatenate the files in the order they're passed at the command line.

If you're using globs or many files, you may start to run into problems with managing dependencies between scripts. In this case, you should use the Closure Library. It contains functions for enforcing dependencies between scripts, and Closure Compiler will re-order the inputs automatically.

How to Contribute

Reporting a bug

1. First make sure that it is really a bug and not simply the way that Closure Compiler works (especially true for ADVANCED_OPTIMIZATIONS).

- Check the official documentation
- Consult the FAQ
- Search on Stack Overflow and in the Closure Compiler Discuss Group

2. If you still think you have found a bug, make sure someone hasn't already reported it. See the list of known issues.

3. If it hasn't been reported yet, post a new issue. Make sure to add enough detail so that the bug can be recreated. The smaller the reproduction code, the better.

Suggesting a Feature

1. Consult the FAQ to make sure that the behaviour you would like isn't specifically excluded (such as string inlining).
2. Make sure someone hasn't requested the same thing. See the list of known issues.
3. Read up on what type of feature requests are accepted.
4. Submit your reqest as an issue.

Submitting patches

1. All contributors must sign a contributor license agreement. See the CONTRIBUTORS file for details.
2. To make sure your changes are of the type that will be accepted, ask about your patch on the Closure Compiler Discuss Group
3. Fork the repository.
4. Make your changes.
5. Submit a pull request for your changes. A project developer will review your work and then merge your request into the project.

Closure Compiler License

Copyright 2009 The Closure Compiler Authors.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Dependency Licenses

Rhino

Code Path	`src/com/google/javascript/rhino`, `test/com/google/javascript/rhino`
URL	http://www.mozilla.org/rhino
Version	1.5R3, with heavy modifications
License	Netscape Public License and MPL / GPL dual license
Description	A partial copy of Mozilla Rhino. Mozilla Rhino is an implementation of JavaScript for the JVM. The JavaScript parse tree data structures were extracted and modified significantly for use by Google's JavaScript compiler.
Local Modifications	The packages have been renamespaced. All code not relevant to the parse tree has been removed. A JsDoc parser and static typing system have been added.

Args4j

Code Path	`lib/args4j.jar`
URL	https://args4j.dev.java.net/

Version	2.0.26
License	MIT
Description	args4j is a small Java class library that makes it easy to parse command line options/arguments in your CUI application.
Local Modifications	None

Guava Libraries

Code Path	`lib/guava.jar`
URL	http://code.google.com/p/guava-libraries/
Version	17.0
License	Apache License 2.0
Description	Google's core Java libraries.
Local Modifications	None

JSR 305

Code Path	`lib/jsr305.jar`
URL	http://code.google.com/p/jsr-305/
Version	svn revision 47
License	BSD License
Description	Annotations for software defect detection.
Local Modifications	None

JUnit

Code Path	`lib/junit.jar`
URL	http://sourceforge.net/projects/junit/
Version	4.11
License	Common Public License 1.0
Description	A framework for writing and running automated tests in Java.
Local Modifications	None

Protocol Buffers

Code Path	`lib/protobuf-java.jar`
URL	http://code.google.com/p/protobuf/

Version	2.5.0
License	New BSD License
Description	Supporting libraries for protocol buffers, an encoding of structured data.
Local Modifications	None

Ant

Code Path	`lib/ant.jar`, `lib/ant-launcher.jar`
URL	http://ant.apache.org/bindownload.cgi
Version	1.8.1
License	Apache License 2.0
Description	Ant is a Java based build tool. In theory it is kind of like "make" without make's wrinkles and with the full portability of pure java code.
Local Modifications	None

JSON

Code Path	`lib/json.jar`
URL	http://json.org/java/index.html
Version	JSON version 20090211
License	MIT license
Description	JSON is a set of java files for use in transmitting data in JSON format.
Local Modifications	None

Mockito

Code Path	`lib/mockito-core.jar`
URL	https://code.google.com/p/mockito
Version	1.9.5
License	MIT license
Description	Mockito is an open source testing framework for Java. The framework allows the creation of Test Double objects (called "Mock Objects") in automated unit tests for the purpose of Test-driven Development (TDD) or Behavior Driven Development (BDD).
Local Modifications	None

Objenesis

Code Path	`lib/objenesis.jar`
URL	http://objenesis.org
Version	1.2
License	Apache 2.0 license
Description	Depended by lib/mockito-core.jar, not used directly.
Local Modifications	None

Node.js Closure Compiler Externs

Code Path	`contrib/nodejs`
URL	https://github.com/dcodeIO/node.js-closure-compiler-externs
Version	e891b4fbcf5f466cc4307b0fa842a7d8163a073a
License	Apache 2.0 license
Description	Type contracts for NodeJS APIs
Local Modifications	Substantial changes to make them compatible with NpmCommandLineRunner.