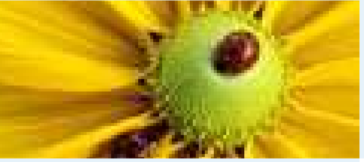
Google's Closure Tools

Tom Payne / github.com/twpayne

29 February 2012

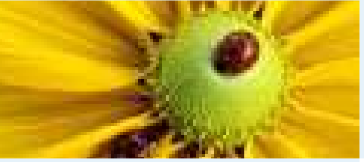


Closure Tools

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Library
- **Compiler**
- Linter
- Templates
- Stylesheets

$\rightarrow$ http://code.google.com/closure/

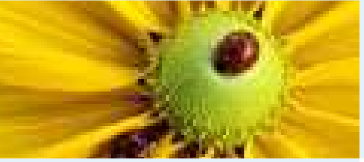


Library

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Extensive
- Modular
- Cross-browser
- Tested
- Well documented
- A “standard library” for Javascript

→ http://code.google.com/closure/library/



Stylesheets and templates

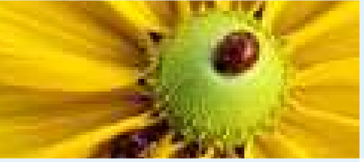
Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

Stylesheets:

- CSS with a pre-processor (*c.f.* less)
- Class renaming
- Optimization / compresssion

Templates:

- Client (JS) and server-side (Java)
- Integrates with CSS class renaming

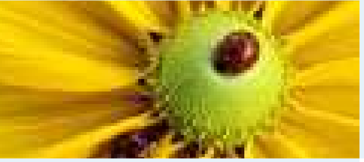


Linter

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Common sources of error
- Coding style

→ http://code.google.com/p/google-styleguide/

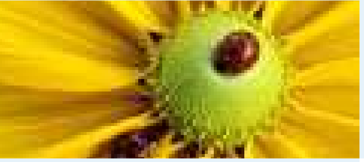


Compiler

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Compiles Javascript to smaller, faster Javascript
- Output is a monolithic Javascript file
- Minimiser
- Optimiser
- Tightly integrated with library

→ http://code.google.com/closure/compiler/



Why compile?

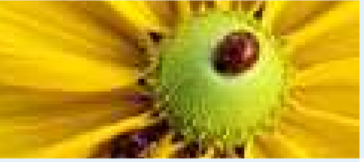
Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

Humans want:

- Code with clear intent
- Well-structured code
- Testing and debugging support

Computers want:

- Compact code
- Code that runs fast
- No unnecessary code

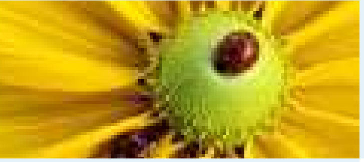


Optimization

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Smaller
- Faster
- Correct
- Compresses well
- Removes dead code

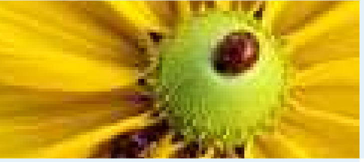
→ http://code.google.com/p/closure-compiler/source/browse/



Compilation levels

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

1. Whitespace only
2. Simple optimizations ($1.25\times$, $1.5\times$ gzip'ed)
3. Advanced optimizations ($4\times$, $8\times$ gzip'ed)

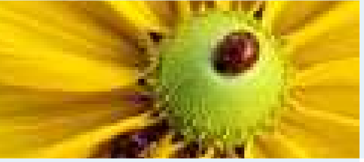


Language extensions

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Uses @jsdoc tags in comments
- Strict, static type checking
- Classical inheritance with constructors and interfaces
- Public, protected and private methods and attributes
- Constants, typedefs and enums
- Pre-processor
- Special treatment of `goog.base`
- No `eval`

→ http://code.google.com/closure/compiler/docs/js-for-compiler.html
→ http://code.google.com/closure/compiler/docs/limitations.html

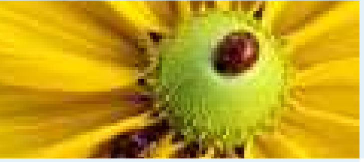


Name mangling

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Internally consistent
- Properties only, not strings
  - `obj.prop = obj['prop'] + 1; // wrong...`
  - `o.p = o['prop'] + 1; // ...when compiled`
- Need to explicitly specify exported symbols ("exports")
- Need to explicitly specify imported symbols ("externs")
- Can write interface files for external libraries

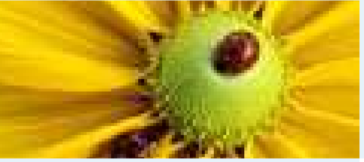
→ http://code.google.com/closure/compiler/docs/api-tutorial3.html



Modules

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- In each source file (module):
  - Declare provides with `goog.provides`
  - Declare requirements with `goog.require`
- Throw everything at `depswriter.py`/`closurebuilder.py`
- Emits only what you need (custom builds :-))

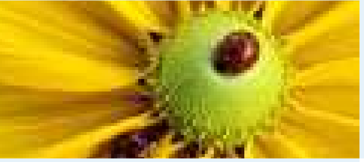


Uncompiled code

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- Load three scripts:
  1. `<script src="closure/goog/base.js">`
  2. `<script src="deps.js">`
  3. `<script>goog.require('my.module');</script>`
- `depswriter.py` generates `deps.js` (the map between modules and source files)
- `goog.require` loads source files as needed
- Great for debugging

→ http://code.google.com/closure/library/docs/depswriter.html

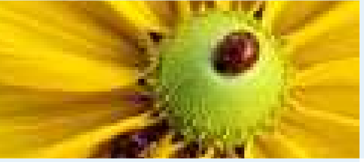


Compiled code

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

- `closurebuilder.py` builds monolithic JS files
- Load one script:
  - `<script src="compiled.js">`
- Pass `--namespace=my.module` to `closurebuilder.py` to set the "main" module
- Hard to debug
- FireBug extension

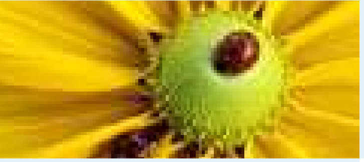
→ http://code.google.com/closure/library/docs/closurebuilder.html
→ http://code.google.com/closure/compiler/docs/inspector.html



Gotchas

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

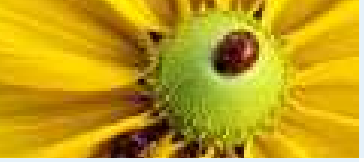
- Name mangling
- Mismatch from imposing strict typing on a dynamic language
- Differences between compiled and uncompiled code
- No `$(document).ready()` by design
- “Clever” Javascript libraries may not be compatible



Practical experience

- Closure Tools
- Library
- Stylesheets and templates
- Linter
- Compiler
- Why compile?
- Optimization
- Compilation levels
- Language extensions
- Name mangling
- Modules
- Uncompiled code
- Compiled code
- Gotchas
- Practical experience
- Demonstration

- Solves many problems with Javascript well
- Catches bugs early - speeds up development
- Tools work best when used together
- Debugging is OK
- Makes Javascript more like Java :-(/ :-)
- Refactoring required to use advanced optimizations
- Long compile times (JVM startup, but multithreaded)
- Needs a good build system
- Interfacing with external packages can be tiresome
- Very effective obfuscator :-)



Demonstration

Closure Tools
Library
Stylesheets and templates
Linter
Compiler
Why compile?
Optimization
Compilation levels
Language extensions
Name mangling
Modules
Uncompiled code
Compiled code
Gotchas
Practical experience
Demonstration

```
git clone https://github.com/twpayne/closure-toy.git
cd closure-toy
make
```