npm install gulp@latest

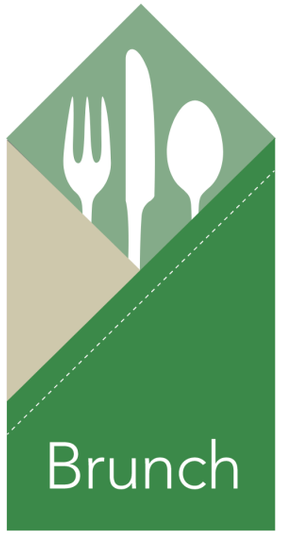
Blaine Bublitz - Lead Maintainer
Berlin Node.js Meetup (May 2019)

What is it?

Build System

GRUNT

BABEL



Concerns

Task System

task(fn)

Composition

series(…fn)

parallel(…fn)

Pipeline

src(globs)

dest(outDir)

.pipe(plugin)*

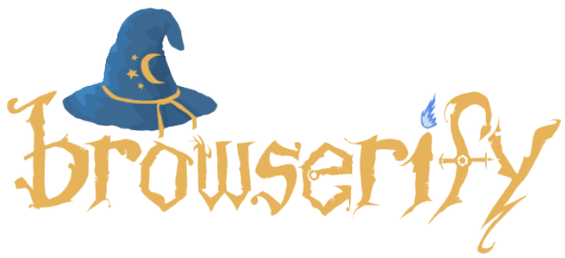
Watching

watch(globs, fn)

***Provided by node.js streams**

Non-Concerns

Bundling



Language Compilation

Gulp is just node - you can bring whatever tools you need for Bundling, Language Compilation, etc

BYOB (bring your own bundler)

Why use it?

Might not need advanced tool

Why use it?

Probably need power tool

Excels at

- Complex serial & parallel compositions
- Applying multiple transforms to many files, without multiple read/writes
- Conditional/phased builds (more on that later)
- Use pre-existing node code and experience

v3.x

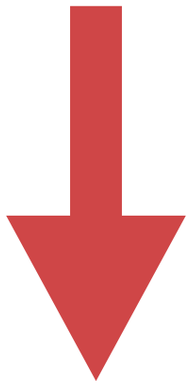
```
gulp.task('clean', function() {
  rimraf.sync('output');
});

gulp.task('html', ['clean'], function() {
  return gulp.src(['input/html/*.pug', '!input/html/_*.pug'])
    .pipe(pug())
    .pipe(gulp.dest('output/html'))
});

gulp.task('css', ['clean'], function() {
  return gulp.src('input/css/*.scss')
    .pipe(sourcemaps.init({ loadMaps: true }))
    .pipe(sass())
    .pipe(minifyCSS())
    .pipe(sourcemaps.write())
    .pipe(gulp.dest('output/css'))
});

gulp.task('js', ['clean'], function() {
  return rollup({
      input: 'input/js/index.js', sourcemap: true, format: 'iife'
    })
    .pipe(source('app.js'))
    .pipe(buffer())
    .pipe(sourcemaps.init({ loadMaps: true }))
    .pipe(babel({ presets: ['@babel/preset-env'] }))
    .pipe(uglify())
    .pipe(sourcemaps.write('./'))
    .pipe(gulp.dest('output/js'))
});

gulp.task('default', ['html', 'css', 'js']);
```

```
function clean(done) {
  rimraf('output', done);
}

function html() {
  return pipeline([
    src(['input/html/*.pug', '!input/html/_*.pug']),
    pug(),
    dest('output/html')
  ]);
}

function css() {
  return pipeline([
    src('input/css/*.scss', { sourcemaps: true }),
    sass(),
    minifyCSS(),
    dest('output/css', { sourcemaps: true })
  ]);
}

function js() {
  return pipeline([
    rollup({
      input: 'input/js/index.js', sourcemap: true, format: 'iife'
    }),
    source('app.js'),
    // Looking to remove these soon (gulpjs/vinyl-sourcemap#32)
    buffer(),
    sourcemaps.init({ loadMaps: true }),
    babel({ presets: ['@babel/preset-env'] }),
    uglify(),
    dest('output/js', { sourcemaps: './' })
  ]);
}

export default series(clean, parallel(html, css, js));
```

What changed?

No more sync tasks

```
gulp.task('clean', function() {
  rimraf.sync('output');
});
```



```
function clean(done) {
  rimraf('output', done);
}
```

Async completion

Previously, we only supported:

- Streams
- Promises (spotty)
- Callbacks

But now, we support:

- Streams
- Promises
- Event Emitters
- Child Processes
- Observables
- Callbacks
- Async/Await

Further reading:

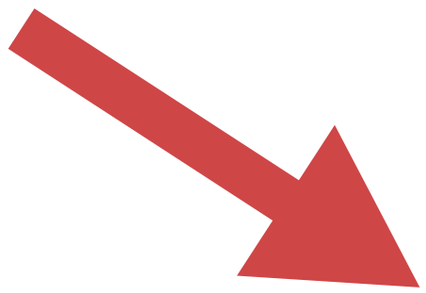
Getting Started guide: Async Completion

What changed?

Named functions

Avoid registering private tasks (clean, html, css, js)

```
gulp.task('clean', function() { ... });
gulp.task('html', ['clean'], function() { ... });
gulp.task('css', ['clean'], function() { ... });
gulp.task('js', ['clean'], function() { ... });
```



```
function clean() { ... }
function html() { ... }
function css() { ... }
function js() { ... }
```

Since we aren't registering private tasks, they don't each need a dependency on "clean"

What changed?

Stream.pipeline instead of pipe*

```
gulp.task('html', ['clean'], function() {
  return gulp.src([
      'input/html/*.pug',
      '!input/html/_*.pug'
    ])
    .pipe(pug())
    .pipe(gulp.dest('output/html'))
});
```

***Not restricted to gulp 4**

```
function html() {
  return pipeline([
    src([
      'input/html/*.pug',
      '!input/html/_*.pug'
    ]),
    pug(),
    dest('output/html')
  ]);
}
```

What changed?

Integrated source map support

```
gulp.src('input/css/*.scss')
  .pipe(sourcemaps.init({ loadMaps: true }))
```

```
.pipe(sourcemaps.write())
.pipe(gulp.dest('output/css'))
```

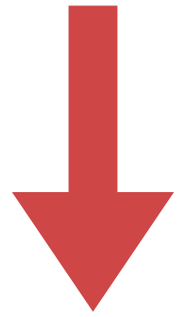
```
src('input/css/*.scss', { sourcemaps: true })
```

```
dest('output/css', { sourcemaps: true })
```

What changed?

Task composition using series & parallel

```
gulp.task('js', ['clean'], function() { ... });
gulp.task('default', ['html', 'css', 'js']);
```



```
series(
  clean,
  parallel(html, css, js)
);
```

What changed?

Exports!*

***Not restricted to gulp 4 but requires gulp-cli**

```
gulp.task('clean', ...);
gulp.task('html', ...);
gulp.task('css', ...);
gulp.task('js', ...);
gulp.task('default', ...);
```

```
module.exports = {
  /* Might not export
     private tasks */
  html, css, js,
  default: series(
    clean,
    parallel(html, css, js)
  )
};
```

```
/* Might not export
   private tasks */
export { html, css, js };
export default series(
  clean,
  parallel(html, css, js)
);
```

gulp-cli

- Separate global install - npm install -g gulp-cli
- Much smaller
- Supports gulp v3 and v4
- Versioned separately
- New features!
  - Task metadata
  - Config files (.gulp.js files)
  - New flags (--verify, --series)

Further reading:

Gulp Sips: New Command Line Interface

Task metadata

Better document tasks for consumers

```
function css() { ... }
css.description = 'Compile & minify SCSS files.';
css.flags = {
  '--inline-sourcemaps': 'Inline sourcemaps in the
output instead of separate files.',
  '--no-minify': 'Avoid the minify step.'
};
```

Demo!

Further reading:

Gulp Sips: Custom Task Metadata

Phased builds

- **src**
  - Readable or Transform stream
  - Usable at beginning or middle
- **dest & symlink**
  - Writeable or Transform streams
  - Usable in the middle or end

```
function css() {
  return pipeline([
    src('input/css/*.scss'),
    sass(),
    dest('output/css'),
    src('vendor/*.css'),
    minifyCSS(),
    rename({ extname: '.min.css' }),
    dest('output/css')
  ]);
}
```

Demo!

Conditional builds

Instead of composing tasks,
pipelines can be varied on conditionals*

*Not restricted to gulp 4

```
function build() {
  return pipeline([
    src('input/{css,img,js}/*'), // Notice no extension?
    gulpIf('*.css', autoprefixer()),
    gulpIf('*.js', uglify()),
    gulpIf(process.env.NODE_ENV === 'production', rev()),
    gulpIf('*.png', symlink('./output'), dest('./output'))
  ]);
}
```

Demo!

Improved watcher

Behavior consistent with task system

- Supports async completion
- Can trigger an initial run
- Delayed by 200ms
- Queues runs

Demo!

Even more new things

- **Documentation website**
  - Getting Started guide
  - Rewritten API docs
- **Gulp Office Hours**
- esm support!
- gulp.symlink()
- A real logger
- Gulp Sips blog posts
- --tasks-json flag
- Custom registries
  - Define custom behavior
  - Share tasks

If something isn’t documented, we’d love for you to contribute!

Thanks!

https://gulpjs.com

https://opencollective.com/gulpjs

gulpjs

BlaineBublitz

gulpjs

phated

Slides and Demos:
https://github.com/phated/talks