

Using Grunt

to Manage Drupal Build and Testing Tools

DrupalCon Los Angeles 2015

5/13/2015

Joe Turgeon

Director of Engineering

Email: jturgeon@phase2technology.com

GitHub: [@arithmetic](https://github.com/arithmetic)

Twitter: [@arithmetic](https://twitter.com/arithmetic)

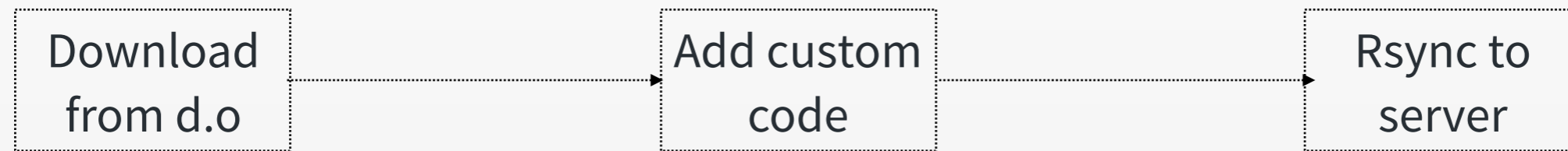


Does Drupal Need a Build Tool?



Drupal development evolved

Install and run approach:



Build and test approach:



Why build Drupal?

- End anti-pattern of mixing core/contrib and custom code
- Improve visibility into projects with a make file
- Simplify use of external libraries and resources
- Enforce reproducibility and automation



Why test Drupal?

- Complex sites need more testing than “clicking around”
- Empower devs to produce quality code
- Find regressions, code style issues before integration



Open questions

- How can we make sure everyone uses the same tools in the same way?
- How can we separate source code from dependencies and build output?



Enter Grunt



Why Grunt?

- JavaScript-based task runner
- Community-supported, widely-adopted, flexible
- Only requirement is Node.js
- Straightforward approach: tasks in JS, config in JSON
- 4,000+ contributed plugins



Grunt, sold!

- Tutorial on writing a Grunt script to minify JavaScript using the “uglify” tool:

<http://gruntjs.com/getting-started>



Introducing Grunt Drupal Tasks

- Grunt plugin for common Drupal build and testing tasks
- Started as R&D project in January 2014
- Since March 2014, used for many client projects
- Released on GitHub in September 2014
- Supports Drupal 7 and 8



It has opinions

- Enforces practice across team and CI env
- Presumes code workflow
 - Verify quality of custom code
 - Assemble core and contrib with Drush make
 - Run tests
- Project scaffolding



It brings greatness together



It stays out of your way

- Sensible defaults that can be overridden
- Manages its own dependencies
- Tools are ready to use and discoverable



Introducing Gadget

- Easy to start a Drupal project with Grunt Drupal Tasks
- Yeoman Generator
- Released on GitHub in April 2015



Getting Started



Dependencies

- Before starting, make sure you have:
 - Node.js (includes npm)
- Recommended for all supporting tools:
 - Bundler, Composer, Ruby, and RubyGems



Install Gadget

- Install Gadget and its dependencies:

```
npm install -g generator-gadget grunt-cli yo
```



Project set up

- Use Gadget to set up a new Drupal project using Grunt Drupal Tasks:

```
yo gadget
```



```
1. node
dc2015$ yo gadget

  _____
 |         |
 |  --(o)--  |
 |         |
 |-----|
 |  ( _'U'_ )  |
 |     A     |
 |     ~     |
 |-----|
 |  . . .  |
 |  |° 'Y'  |
 |         |

Welcome to Gadget, the
gnarly generator for
Grunt Drupal Tasks!

? Which version of Drupal core would you like to use? (Use arrow keys)
> Drupal 8
  Drupal 7
```




```
1. node
├─ grunt-notify@0.4.1 (stack-parser@0.0.1, which@1.0.9, semver@4.3.3)
└─ grunt-contrib-compress@0.13.0 (prettysize@0.0.3, chalk@0.5.1, archiver@0.13.1)

Installed version 0.6.0 of Grunt Drupal Tasks.

Setting up Drush makefile to install Drupal version 8.0.0-beta10.

create package.json
create Gruntconfig.json
create Gruntfile.js
create composer.json
create gitignore
create phpmd.xml
create src/modules/.gitkeep
create src/profiles/.gitkeep
create src/project.make
create src/sites/default/.gitkeep
create src/static/.gitkeep
create src/themes/.gitkeep
create test/behat.yml
create test/features/bootstrap/FeatureContext.php
create test/features/example.feature
create .gitignore
```



```
1. bash
SOLINK_MODULE(target) Release/contextify.node: Finished
zombie@2.5.1 node_modules/zombie
├─ ms@0.7.1
├─ lazybird@1.0.0
├─ mime@1.3.4
├─ debug@2.1.3 (ms@0.7.0)
├─ iconv-lite@0.4.8
├─ tough-cookie@0.12.1 (punycode@1.3.2)
├─ bluebird@2.9.25
├─ eventsource@0.1.6 (original@0.0.8)
├─ request@2.55.0 (caseless@0.9.0, json-stringify-safe@5.0.0, aws-sign2@0.5.0,
forever-agent@0.6.1, stringstream@0.0.4, oauth-sign@0.6.0, tunnel-agent@0.4.0, i
sstream@0.1.2, node-uuid@1.4.3, qs@2.4.1, combined-stream@0.0.7, form-data@0.2.0
, bl@0.9.4, mime-types@2.0.10, http-signature@0.10.1, hawk@2.3.1, har-validator@
1.7.0)
├─ ws@0.6.5 (options@0.0.6, ultron@1.0.1, nan@1.4.3)
├─ jsdom@1.4.0 (xmlhttprequest@1.7.0, browser-request@0.3.3, cssom@0.3.0, nwmatt
cher@1.3.4, parse5@1.4.2, htmlparser2@3.8.2, cssstyle@0.2.24, contextify@0.1.13)

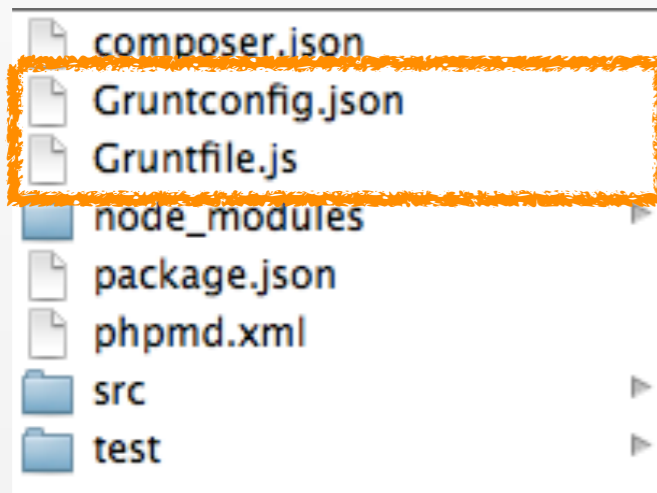
Gadget has finished setting up the Drupal project scaffold with Grunt Drupal Tas
ks!

Run `grunt` to run the first build of this project.

dc2015$
```



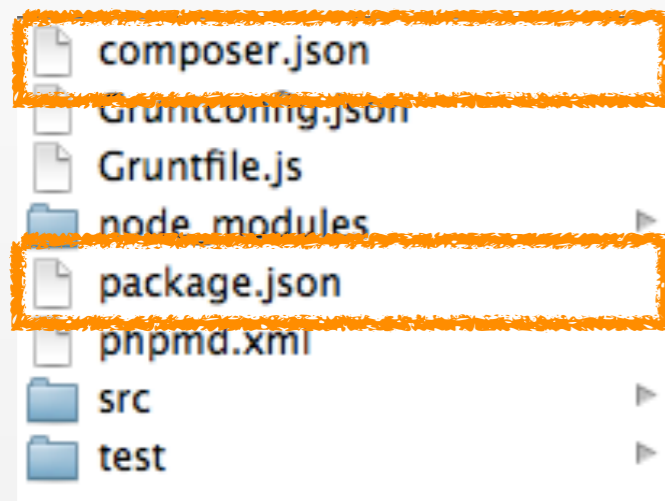
Project scaffolding



GDT project script
and config



Project scaffolding



Defines GDT and project dependencies



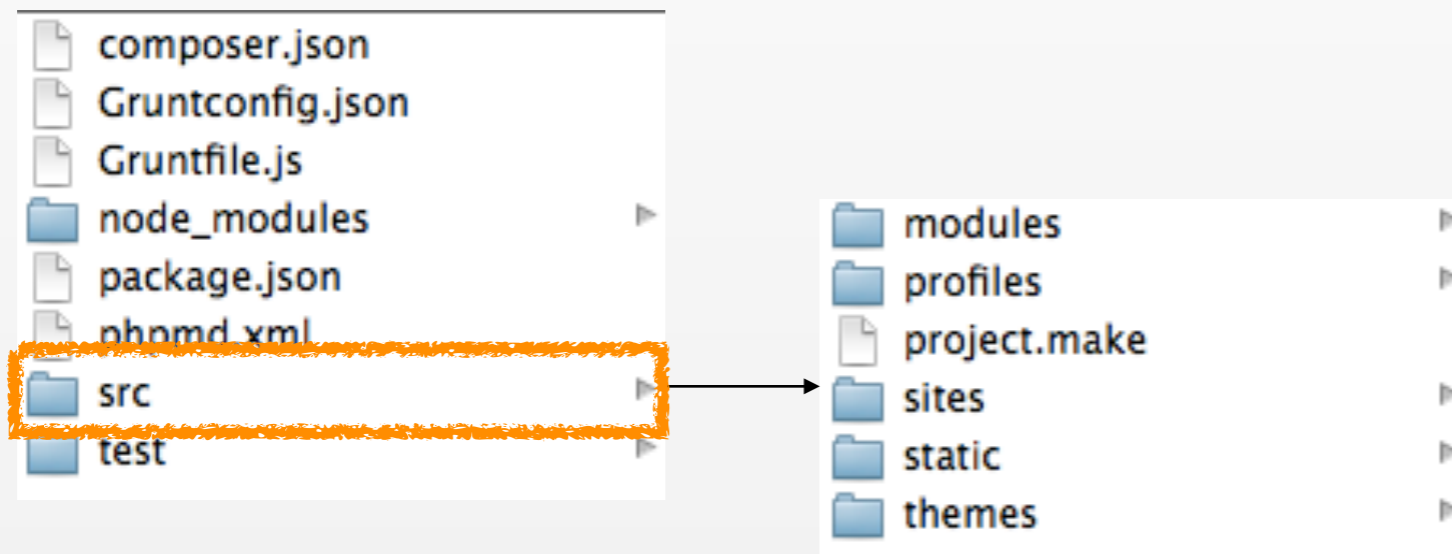
Project scaffolding



GDT
dependencies



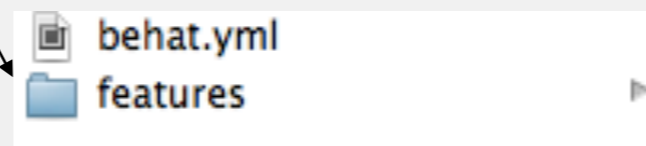
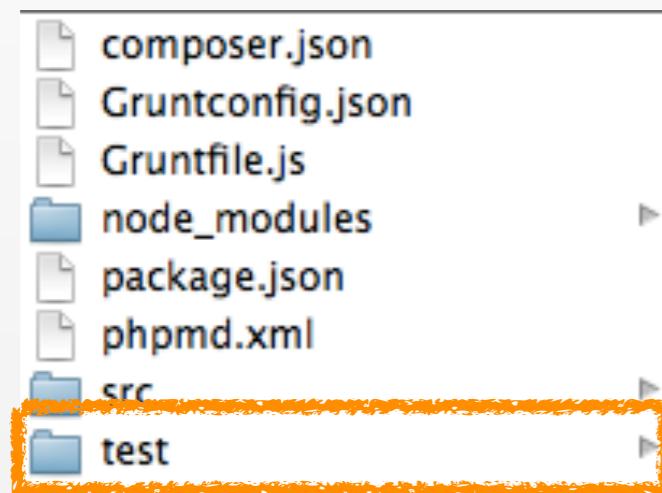
Project scaffolding



Custom Drupal
code and Drush
make file



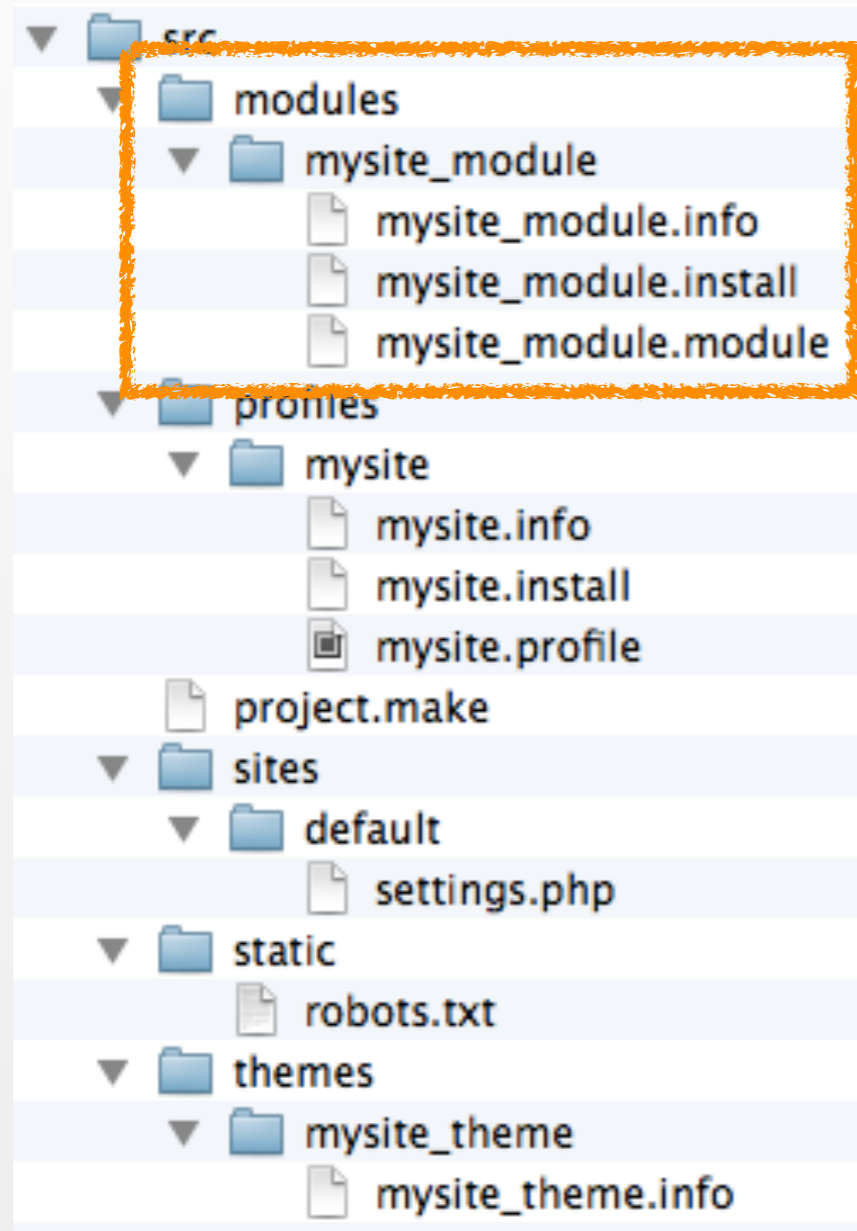
Project scaffolding



Test cases and
config



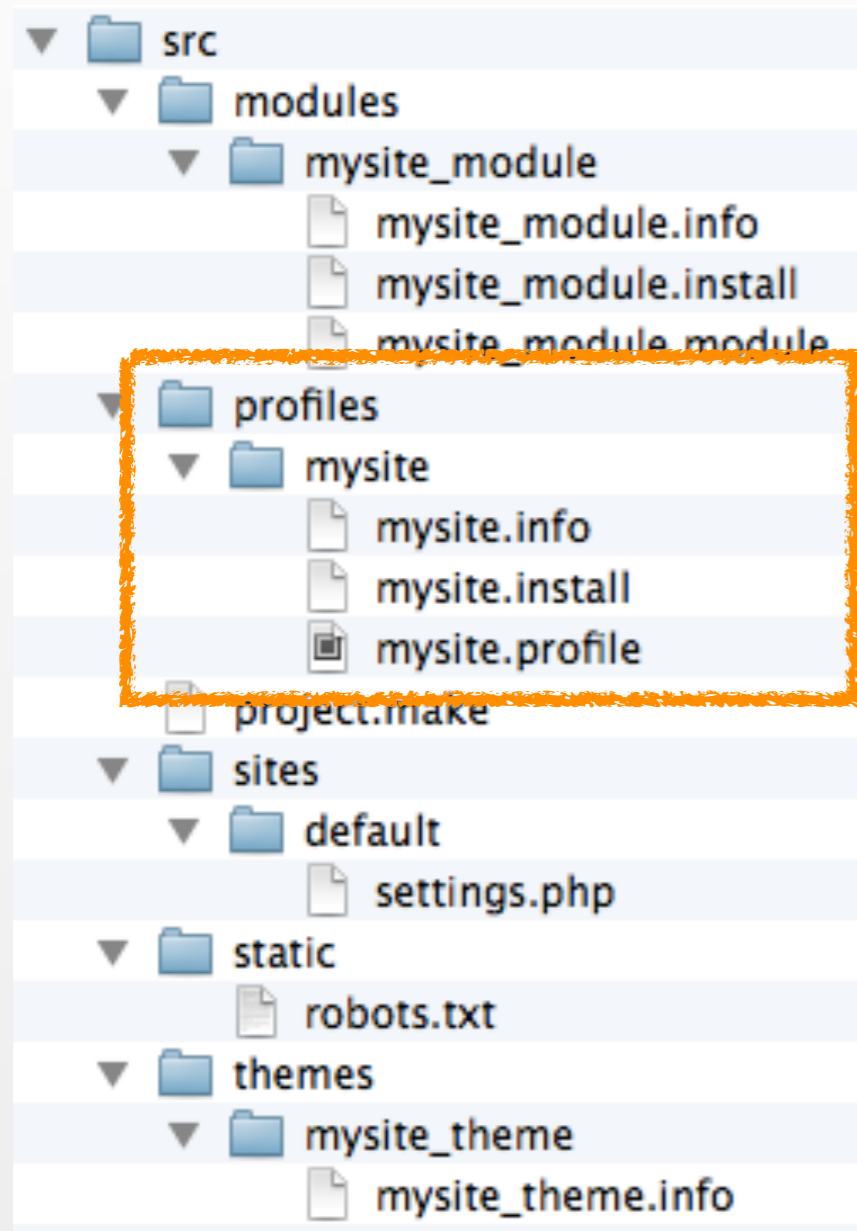
Everything in it's right place



- Modules



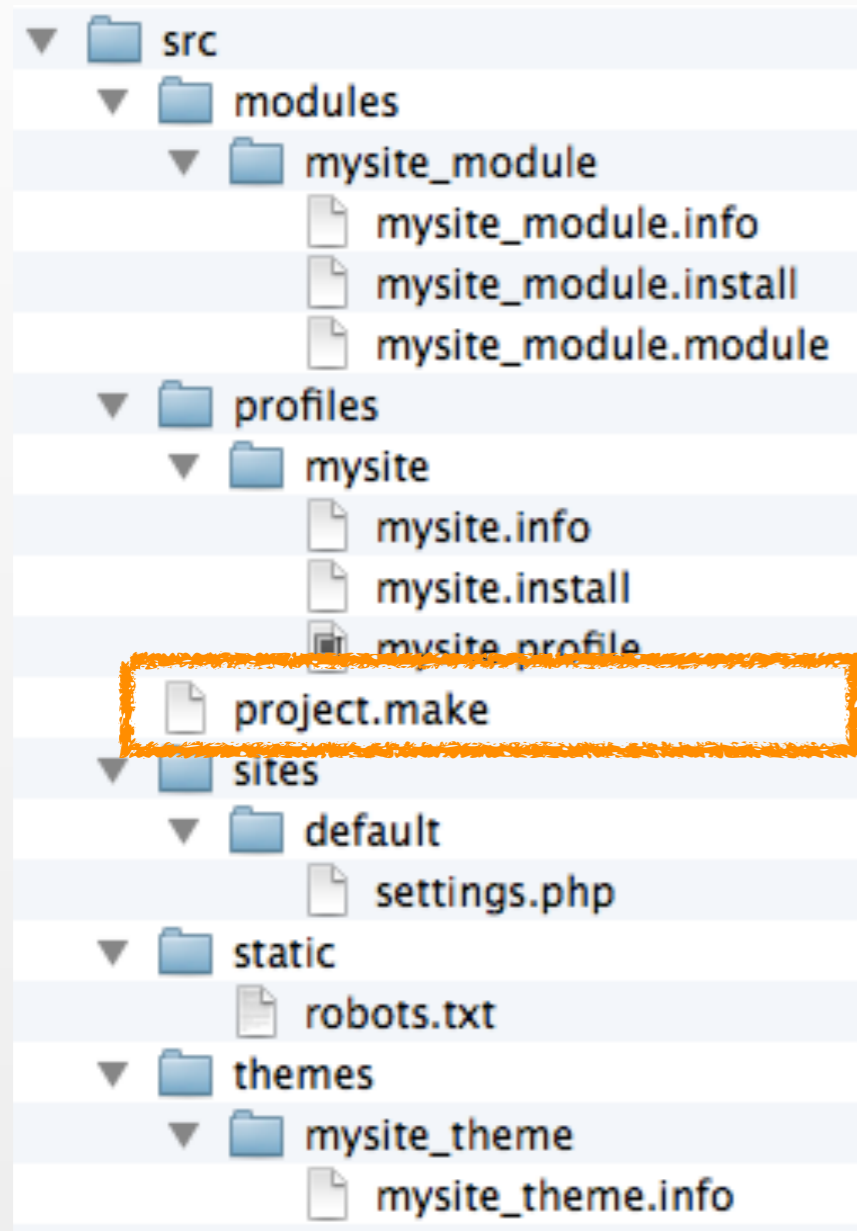
Everything in it's right place



- Modules
- Install profiles



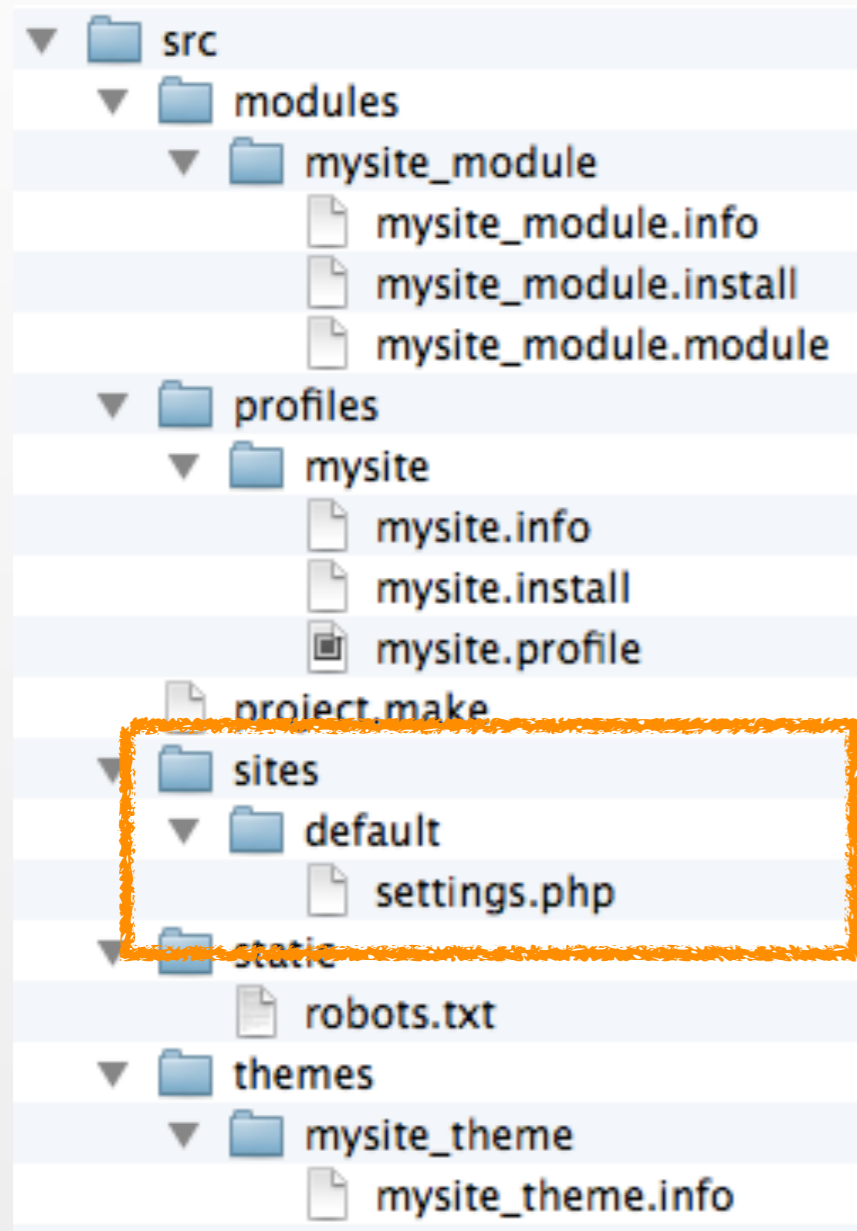
Everything in it's right place



- Modules
- Install profiles
- Drush make file



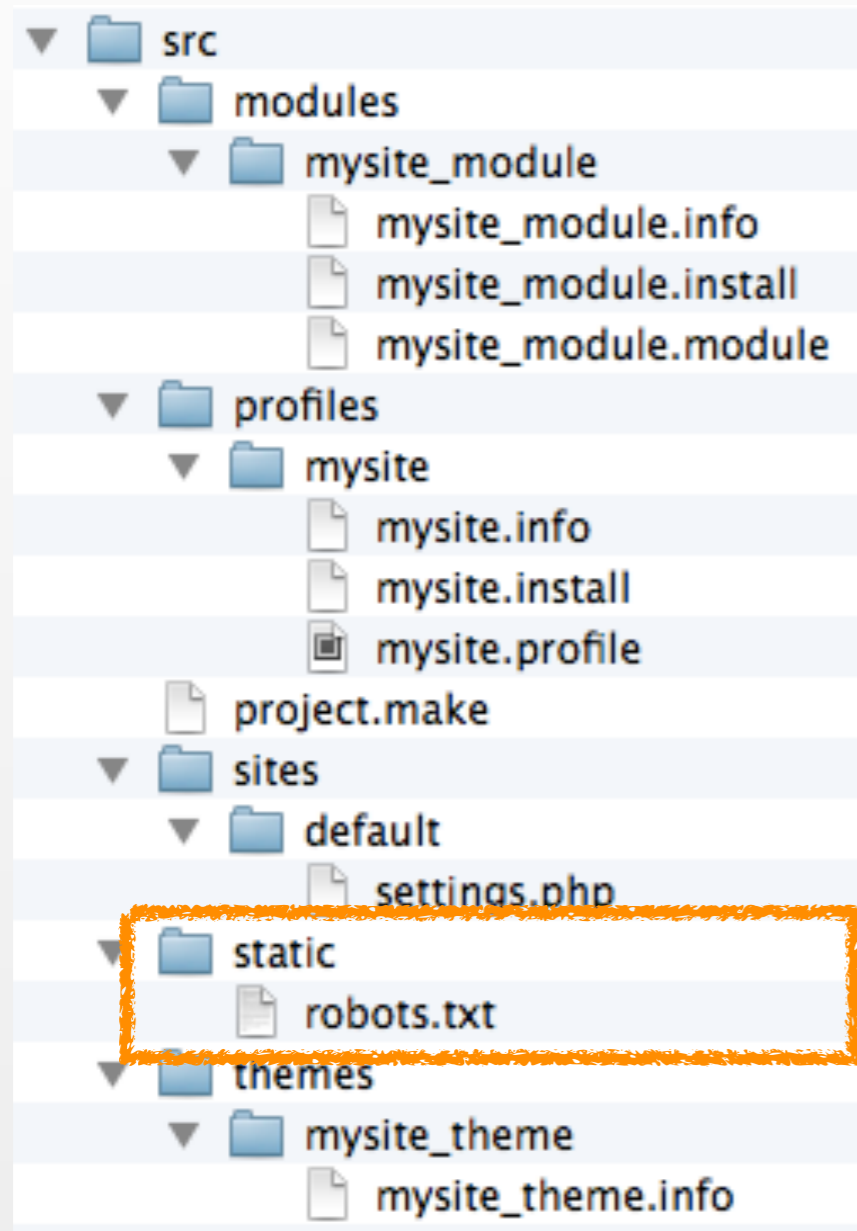
Everything in it's right place



- Modules
- Install profiles
- Drush make file
- Single-/multi-site config



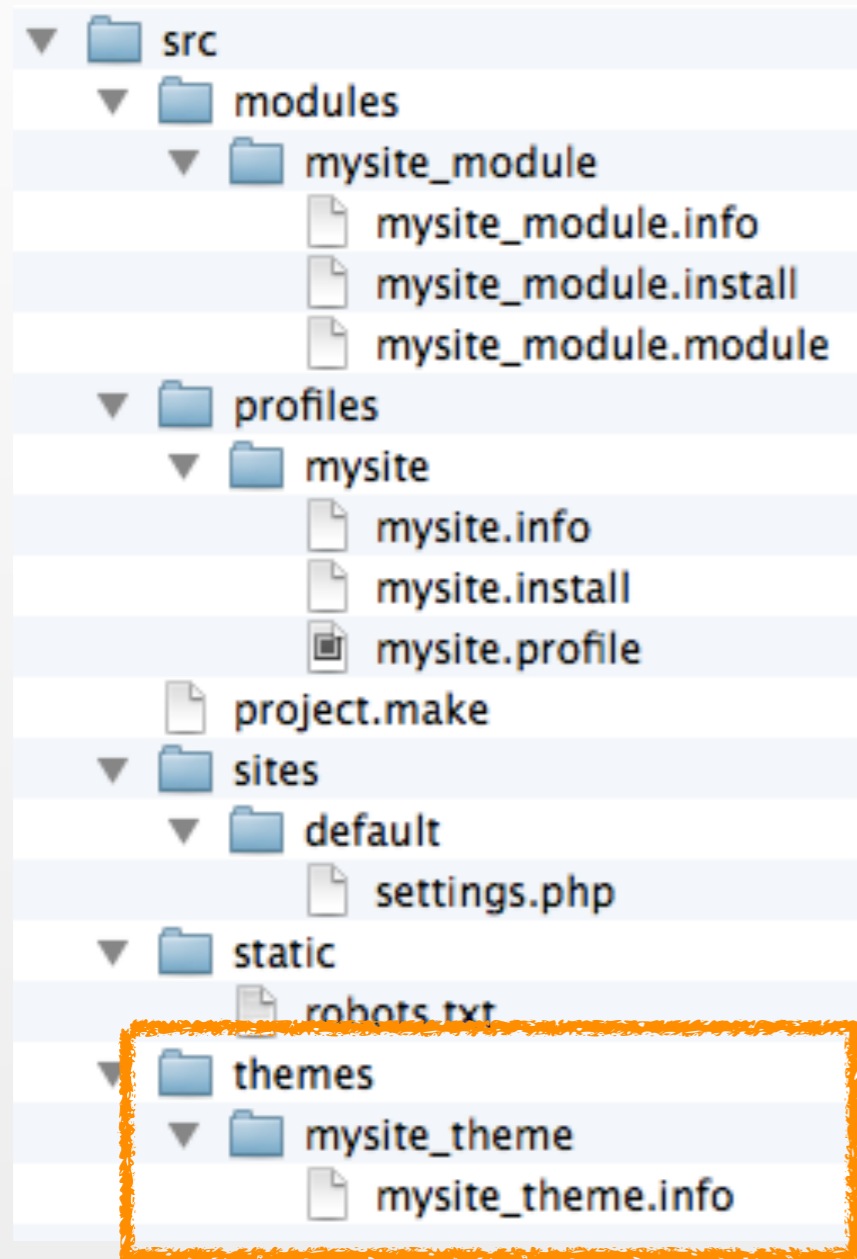
Everything in it's right place



- Modules
- Install profiles
- Drush make file
- Single-/multi-site config
- Static file replacements



Everything in it's right place



- Modules
- Install profiles
- Drush make file
- Single-/multi-site config
- Static file replacements
- Themes



Goals

- Accommodate (and isolate) custom code, Drupal core and contrib code, site settings files, and supporting tools
- Allow tools to work together with minimal glue
- Standardize project code base structure



Building



Project build

- Use Grunt to build the project:

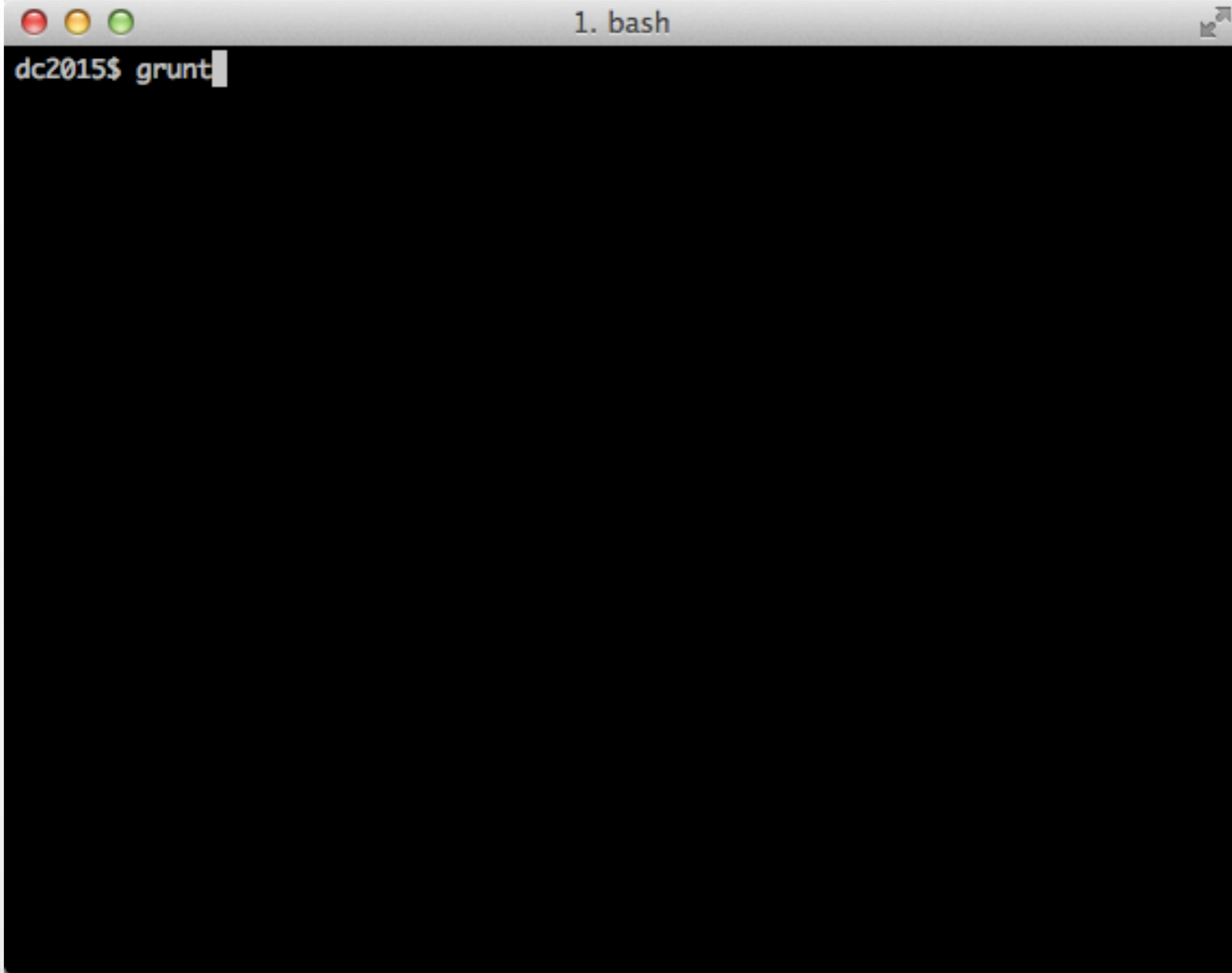
```
grunt
```



Default build steps

- Composer/Bundler ** = if applicable/needed*
- Validate code
- Drush make
- Scaffold (injects custom files into build output)
- Compile theme





A terminal window with a title bar containing three colored window control buttons (red, yellow, green) on the left and the text "1. bash" on the right. The terminal content shows the prompt "dc2015\$" followed by the command "grunt" and a white cursor block at the end of the line.

```
dc2015$ grunt
```



```
1. sh
dc2015$ grunt
Running "notify_hooks" task

Running "composer:install" (composer) task
Warning: This development build of composer is over 30 days old. It is recommend
ed to update it by running "/Users/Turgeon/bin/composer self-update" to get the
latest version.
Loading composer repositories with package information
Installing dependencies (including require-dev)
- Installing symfony/process (v2.6.6)
  Loading from cache

- Installing symfony/css-selector (v2.6.6)
  Loading from cache

- Installing behat/mink (v1.6.1)
  Loading from cache

- Installing behat/mink-zombie-driver (v1.2.0)
  Loading from cache

- Installing symfony/var-dumper (v2.6.3)
  Loading from cache

- Installing d11wtq/boris (v1.0.8)
```




```
1. bash
Running "drush:make" (drush) task
Beginning to build src/project.make. [ok]
drupal-8.0.0-beta10 downloaded. [ok]
  >> Project devel contains 4 modules: kint, devel_node_access, devel_generate, d
  devel.
  >> devel-8.x-1.x-dev downloaded. [ok]
  >> bootstrap-8.x-3.x-dev downloaded. [ok]

Running "clean:default" (clean) task
>> 1 path cleaned.

Running "copy:tempbuild" (copy) task
Created 3285 directories, copied 12710 files

Running "clean:temp" (clean) task
>> 1 path cleaned.

Running "newer-postrun:drushmake:default:1:/Users/Turgeon/work/automation/grush/
dc2015/node_modules/grunt-drupal-tasks/node_modules/grunt-newer/.cache" (newer-p
ostrun) task

Running "scaffold" task
Detected Drupal 8 codebase.

Running "symlink:profiles" (symlink) task
```



```
1. bash
>> Created 0 symbolic links.

Running "symlink:modules" (symlink) task
>> Created 1 symbolic links.

Running "symlink:themes" (symlink) task
>> Created 1 symbolic links.

Running "copy:defaults" (copy) task
Copied 2 files

Running "clean:sites" (clean) task
>> 1 path cleaned.

Running "symlink:sites" (symlink) task
>> Created 1 symbolic links.

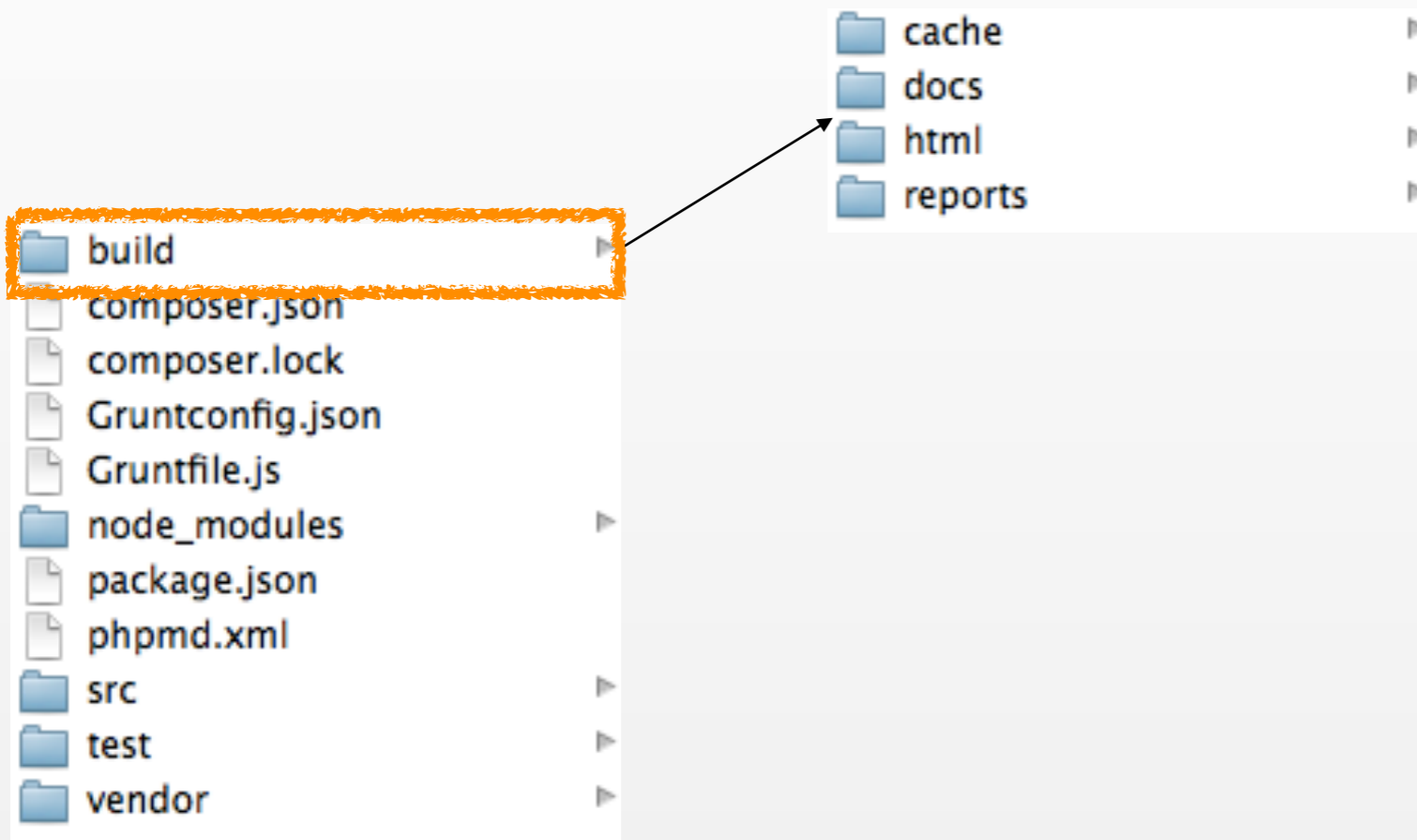
Running "mkdir:files" (mkdir) task
Creating "build/html/sites/default/files"...OK

Running "copy:static" (copy) task
Created 1 directory, copied 1 file

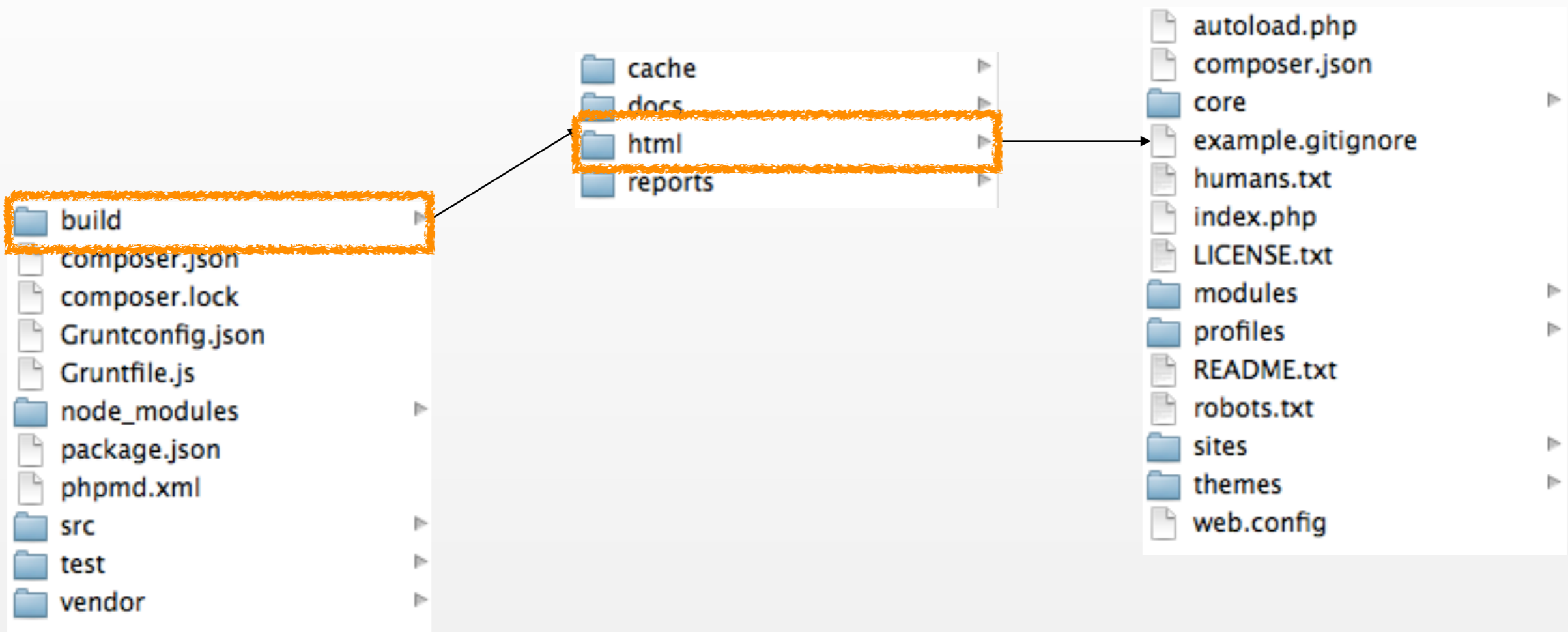
Done, without errors.
dc2015$
```



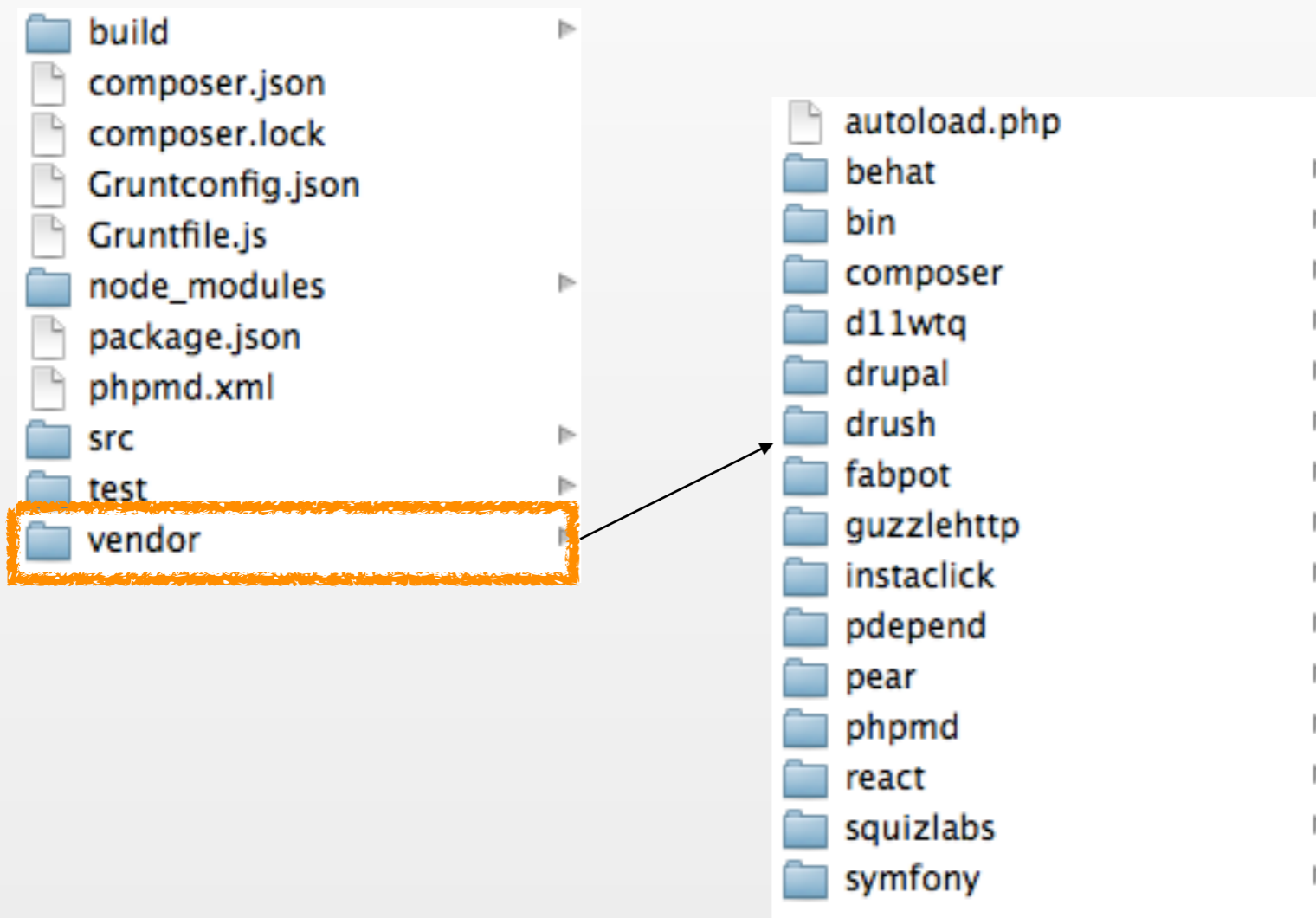
Build output



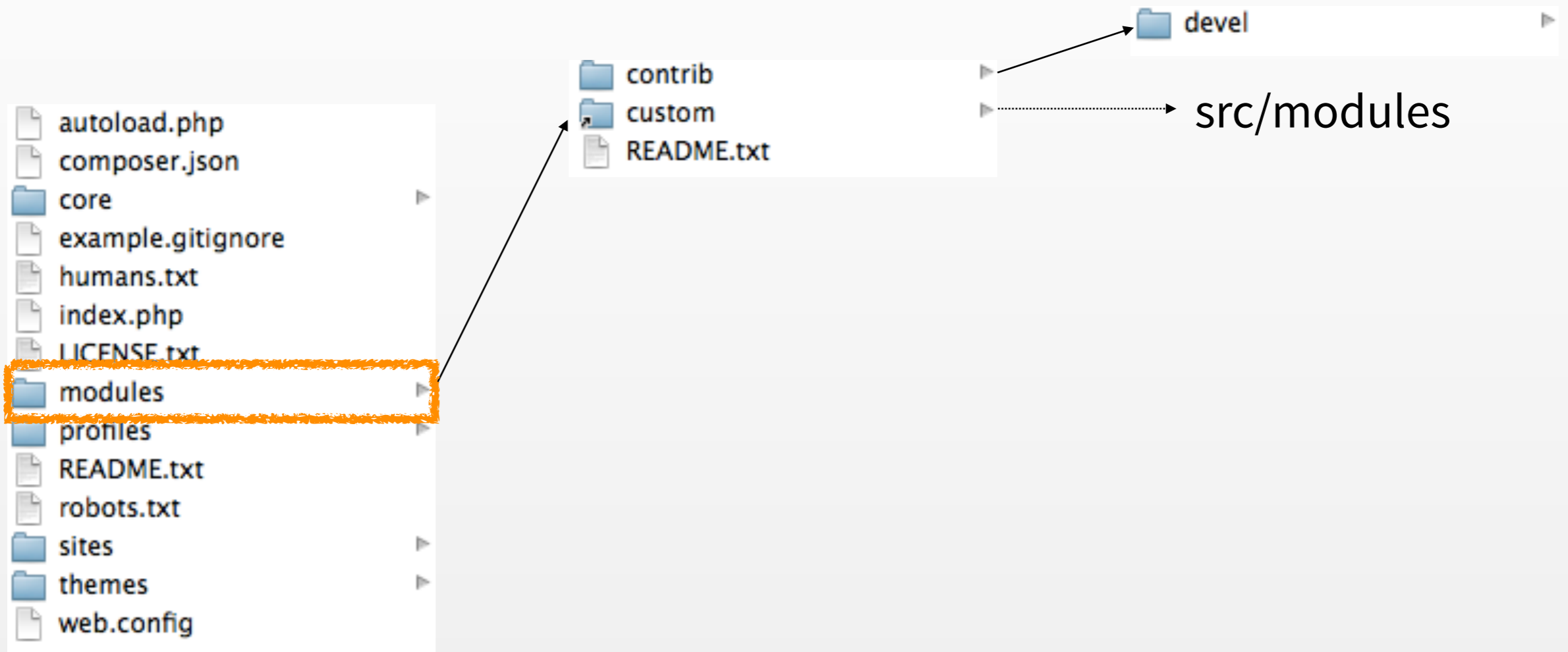
Build output



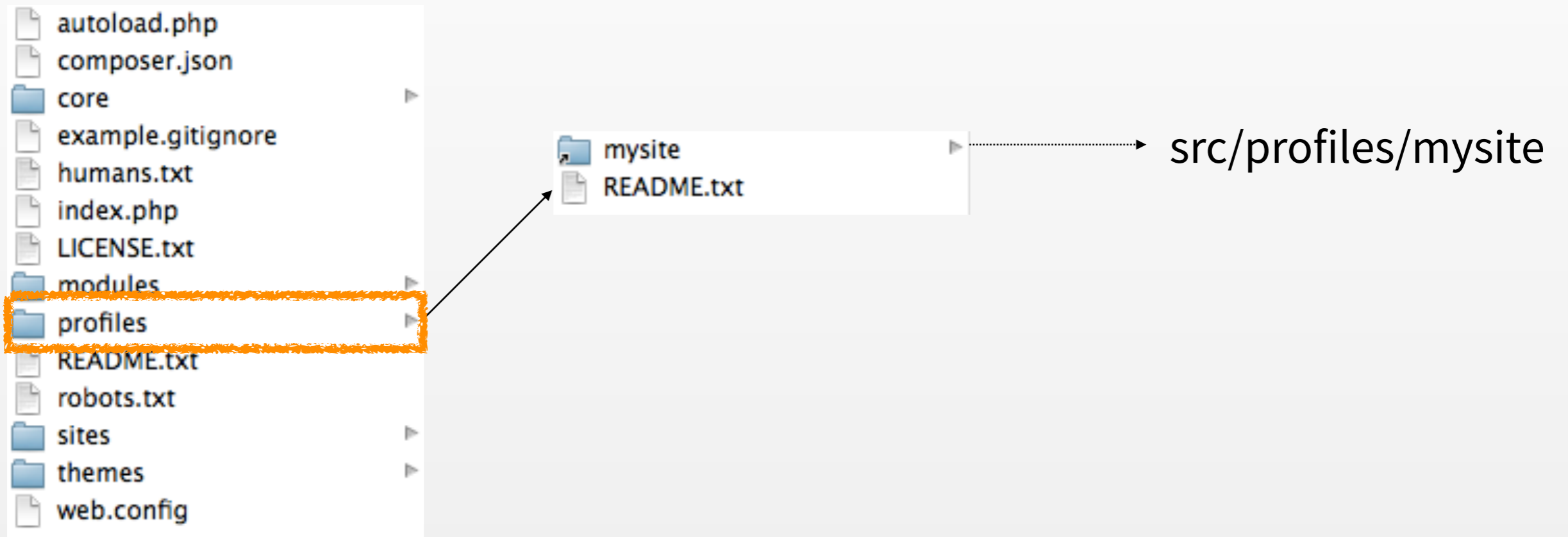
Build output



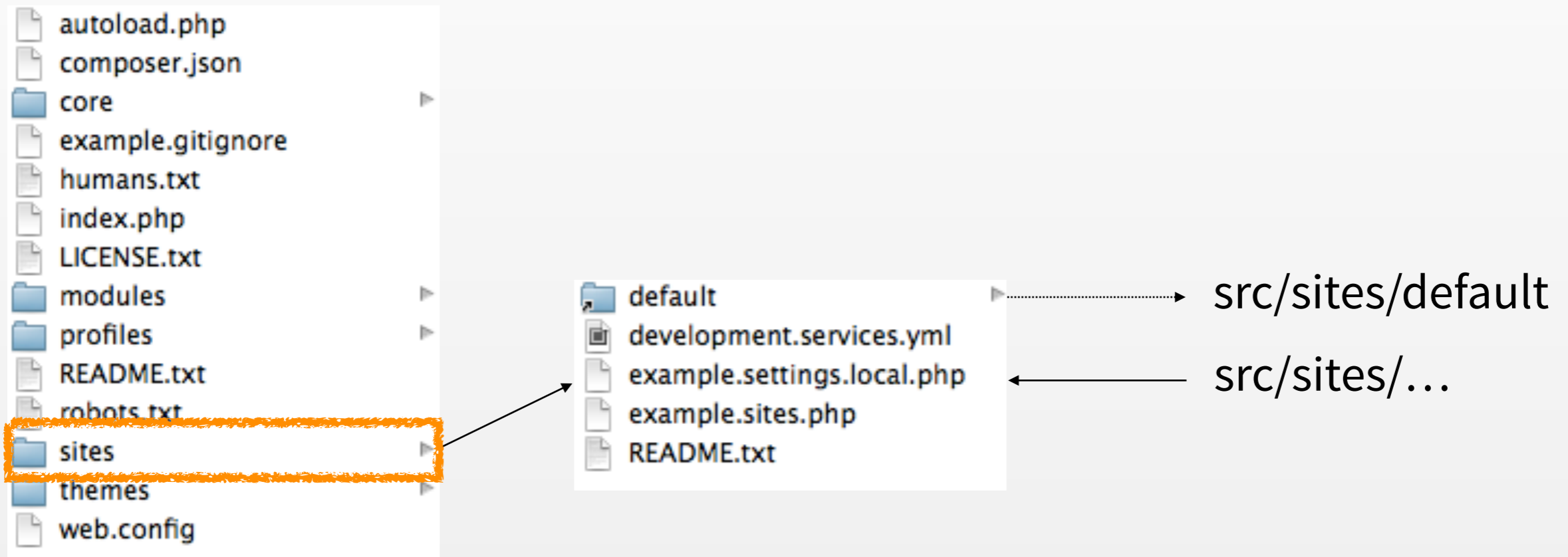
Into build/html/



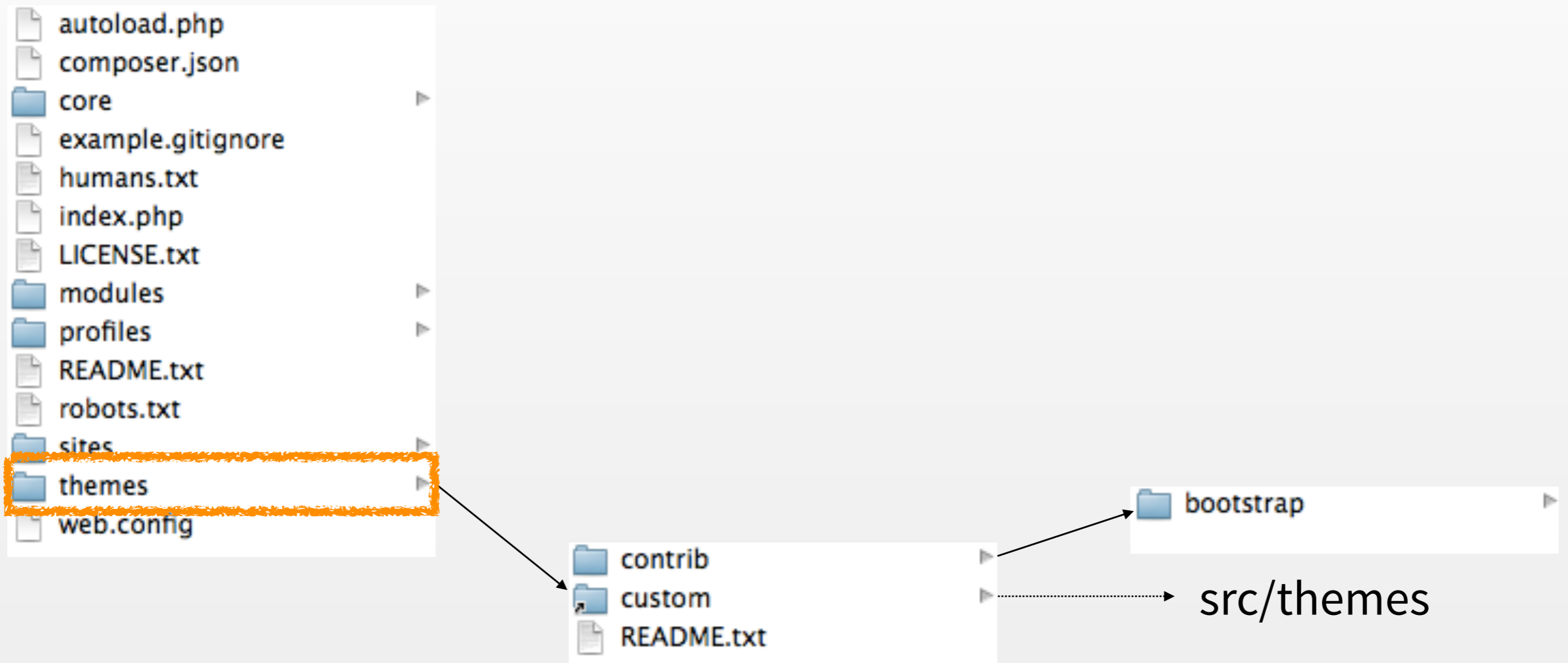
Into build/html/



Into build/html/



Into build/html/



Symlinks, really?

- No need to re-build after each change to a source file
- Build tasks ensure the links are set up correctly
- Relative links within the scaffold structure are portable
- XDebug works across symlinks
- Windows users: Beware! Copy alternative in the backlog



Running automatically

- Runs validate, theme compile, drush make (if needed) whenever file changes are detected:

```
grunt watch
```



```
1. node
dc2015$ grunt watch
Running "notify_hooks" task

Running "watch" task
Waiting...
█
```





1. node



```
dc2015$ grunt watch
```

```
Running "notify_hooks" task
```

```
Running "watch" task
```

```
Waiting...
```

```
>> File "src/modules/mysite_module/mysite_module.module" changed.
```

```
█
```



```
1. node
dc2015$ grunt watch
Running "notify_hooks" task

Running "watch" task
Waiting...
>> File "src/modules/mysite_module/mysite_module.module" changed.
Running "notify_hooks" task

Running "phplint:all" (phplint) task
>> 4 files php linted.

Running "phpcs:validate" (phpcs) task
Starting phpcs (target: validate) in src/**/*.css src/**/*.php src/**/*.module s
rc/**/*.inc src/**/*.install src/**/*.profile !src/sites/** !src/**/*.box.inc !s
rc/**/*.features.*inc !src/**/*.pages_default.inc !src/**/*.panelizer.inc !src/*
/**/*.strongarm.inc

FILE: .../automation/grush/dc2015/src/modules/mysite_module/mysite_module.module
-----
FOUND 3 ERROR(S) AFFECTING 3 LINE(S)
-----
 2 | ERROR | Missing file doc comment
 3 | ERROR | Missing function doc comment
 4 | ERROR | Line indented incorrectly; expected 2 spaces, found 0
-----
```



```
1. node
FOUND 0 ERROR(S) AND 1 WARNING(S) AFFECTING 1 LINE(S)
-----
1 | WARNING | No PHP code was found in this file and short open tags are not
  |         | allowed by this install of PHP. This file may be using short
  |         | open tags but PHP does not allow them.
-----
UPGRADE TO PHP_CODESNIFFER 2.0 TO FIX ERRORS AUTOMATICALLY
-----

FILE: ...Turgeon/work/automation/grush/dc2015/src/profiles/mysite/mysite.profile
-----
FOUND 0 ERROR(S) AND 1 WARNING(S) AFFECTING 1 LINE(S)
-----
1 | WARNING | No PHP code was found in this file and short open tags are not
  |         | allowed by this install of PHP. This file may be using short
  |         | open tags but PHP does not allow them.
-----
UPGRADE TO PHP_CODESNIFFER 2.0 TO FIX ERRORS AUTOMATICALLY
-----

Done, without errors.
Completed in 1.976s at Tue May 12 2015 14:48:37 GMT-0700 (PDT) - Waiting...
█
```



The way of the build

- Captures the daily development workflow
- Produces a complete Drupal site docroot from custom source and a make file
- Build locally, build on integration



Have it your way

- Build tasks can be run individually
- Let's look at the core tasks



Validating



Validating code quality

- Uses lints and static code analysis to quantify code quality
- Run PHP Lint and PHP Code Sniffer (with Drupal coding standards) with:

```
grunt validate
```



PHP Lint

- Verifies syntax with PHP's built-in lint tool

```
1. bash
dc2015$ grunt validate
Running "notify_hooks" task

Running "phplint:all" (phplint) task
Warning: PHP Parse error:  parse error, expecting `"variable (T_VARIABLE)"' or `
'$'` in src/modules/mysite_module/mysite_module.module on line 4
Use --force to continue.

Aborted due to warnings.
dc2015$
```



PHP Code Sniffer

- Verifies code style according to Drupal's code standards

```
1. bash
Running "phpcs:validate" (phpcs) task
Starting phpcs (target: validate) in src/**/*.css src/**/*.php src/**/*.module s
rc/**/*.inc src/**/*.install src/**/*.profile !src/sites/** !src/**/*.box.inc !s
rc/**/*.features.*inc !src/**/*.pages_default.inc !src/**/*.panelizer.inc !src/*
/**/*.strongarm.inc

FILE: ../automation/grush/dc2015/src/modules/mysite_module/mysite_module.module
-----
FOUND 5 ERROR(S) AFFECTING 4 LINE(S)
-----
 2 | ERROR | Missing file doc comment
 3 | ERROR | Missing function doc comment
 4 | ERROR | Line indented incorrectly; expected 2 spaces, found 0
 4 | ERROR | Calls to inbuilt PHP functions must be lowercase; expected "fopen"
   |       | but found "fOpen"
 5 | ERROR | Line indented incorrectly; expected 2 spaces, found 0
-----
UPGRADE TO PHP_CODESNIFFER 2.0 TO FIX ERRORS AUTOMATICALLY
-----
```



Check yo' self

- Encourage devs to validate code before committing
- Focus peer code review on architecture
- Fail integration builds or reject pull requests at certain thresholds



Compiling Themes



Front end tools

- Compass was one of the first non-PHP tools to enter common Drupal practice
- Configure GDT to define themes and compilation approach
- Run individually with ``grunt compile-theme``



Compass

- Include a Gemfile to install Compass and required gems
- Compass compile is run during build

```
1. bash
dc2015$ grunt compile-theme
Running "notify_hooks" task

Running "compass:example_theme" (compass) task
  write src/themes/example_theme/stylesheets/ie.css (0.015s)
  write src/themes/example_theme/stylesheets/print.css (0.001s)
  write src/themes/example_theme/stylesheets/screen.css (0.01s)

Done, without errors.
dc2015$
```



Grunt-to-Grunt

- GDT will be able to delegate theme compilation to Grunt-powered themes
- Theme-supplied Grunt scripts are run during build for any configured themes



Testing



Testing site features

- Provides Behat, the Drupal Extension, other dependencies
- Testing with Behat requires an installed Drupal site accessible by URL
- Two options for running tests locally
 - Using a local env (VM, Docker, W/MAMP)
 - Using PHP/drush's local test server



```
1. bash
dc2015$ grunt drush:liteinstall
Running "notify_hooks" task

Running "drush:liteinstall" (drush) task
You are about to DROP all tables in your '/Users/Turgeon/work/automation/grush/dc2015/build/drupal.sqlite' database. Do you want to continue? (y/n): y
Starting Drupal installation. This takes a while. Consider using the [ok]
--notify global option.
Installation complete. User name: admin User password: KN5tTXpasU [ok]
Congratulations, you installed Drupal! [status]

Done, without errors.
dc2015$
```



```
1. node
dc2015$ grunt drush:liteinstall
Running "notify_hooks" task

Running "drush:liteinstall" (drush) task
You are about to DROP all tables in your '/Users/Turgeon/work/automation/grush/dc2015/build/drupal.sqlite' database. Do you want to continue? (y/n): y
Starting Drupal installation. This takes a while. Consider using the [ok]
--notify global option.
Installation complete. User name: admin User password: KN5tTXpasU [ok]
Congratulations, you installed Drupal! [status]

Done, without errors.
dc2015$ grunt drush:runserver
```



```
1. node
dc2015$ grunt drush:liteinstall
Running "notify_hooks" task

Running "drush:liteinstall" (drush) task
You are about to DROP all tables in your '/Users/Turgeon/work/automation/grush/dc2015/build/drupal.sqlite' database. Do you want to continue? (y/n): y
Starting Drupal installation. This takes a while. Consider using the [ok]
--notify global option.
Installation complete. User name: admin User password: KN5tTXpasU [ok]
Congratulations, you installed Drupal! [status]

Done, without errors.
dc2015$ grunt drush:runserver
Running "notify_hooks" task


Running "drush:runserver" (drush) task
HTTP server listening on 127.0.0.1, port 8080 (see http://127.0.0.1:8080/), serving site default, logged in as admin...
PHP 5.4.30 Development Server started at Tue May 12 15:26:30 2015
Listening on http://127.0.0.1:8080
Document root is /Users/Turgeon/work/automation/grush/dc2015/build/html
Press Ctrl-C to quit.
█
```



Welcome to Site-Install | Site-Install x

127.0.0.1:8080

Joe



Site-Install

Home

User login

Username *


Password *

Log in

- [Create new account](#)
- [Reset your password](#)

Welcome to Site-Install

No front page content has been created yet.



Running tests

- Run tests with:

```
grunt test
```



```
1. node
dc2015$ grunt test
Running "notify_hooks" task

Running "behat:site-default" (behat) task

Found 1 feature file(s). Running 5 at a time.
Started: vendor/bin/behat -c ./test/behat.yml --tags ~@wip ../test/features/example.feature
█
```



```
1. bash
dc2015$ grunt test
Running "notify_hooks" task

Running "behat:site-default" (behat) task

Found 1 feature file(s). Running 5 at a time.
Started: vendor/bin/behat -c ./test/behat.yml --tags ~@wip ../test/features/example.feature

stdout:
Feature: Behat tests for a clean install of the Drupal 8 standard profile. These tests are meant both to verify the Drupal install as well as Behat test features (like API access and JavaScript handling).

Scenario: Ensure the Login link is available for anonymous users. # features/example.feature:6
  Given I am an anonymous user # Drupal\DrupalExtension\Context\DrupalContext::assertAnonymousUser()
  When I am on the homepage # Drupal\DrupalExtension\Context\MinkContext::iAmOnHomepage()
  Then I should see an "input#edit-name" element # Drupal\DrupalExtension\Context\MinkContext::assertElementOnPage()
  And I should see an "input#edit-pass" element # Drupal\DrupalExtension\Context\MinkContext::assertElementOnPage()
```



```
1. bash

    menu item is clicked.
    Given I am logged in as a user with the "administrator" role
# Drupal\DrupalExtension\Context\DrupalContext::assertAuthenticatedByRole()
    And I am on the homepage
# Drupal\DrupalExtension\Context\MinkContext::iAmOnHomepage()
    And I click "My account"
# Drupal\DrupalExtension\Context\MinkContext::assertClick()
    And I click "Edit"
# Drupal\DrupalExtension\Context\MinkContext::assertClick()
    When I fill in "pass[pass1]" with "123"
# Drupal\DrupalExtension\Context\MinkContext::fillField()
    Then I should see "Password strength" in the ".form-item-pass-pass1" element
# Drupal\DrupalExtension\Context\MinkContext::assertElementContainsText()

3 scenarios (3 passed)
14 steps (14 passed)
0m39.66s (40.28Mb)

>> Completed: ./test/features/example.feature - 3 scenarios (3 passed) in 0m39.6
6s (40.28Mb)

Finished in 0m40s

Done, without errors.
dc2015$ █
```



Towards BDD

- GDT supports a TDD/BDD workflow
 - Write tests first
 - Code iteratively with test feedback
 - Automate using “watch” tasks
- Includes tools and test scripts to kickstart testing



Packaging and Deployment



Packaging and deployment

- GDT and scaffolding are meant for local and integration envs
- Prepares code for release
- Hand off to other tools for deployment



Packaging

- Produces a standalone, deployable Drupal docroot
- Excludes supporting tools and config
- Run with `grunt package`

```
1. bash
dc2015$ grunt package
Running "notify_hooks" task

Running "package:tarball" (package) task

Running "compress:package" (compress) task
Created build/packages/package.tgz (10638474 bytes)

Done, without errors.
dc2015$
```



Rolling a release

- Exploring how GDT can support the release process
- Planned approaches
 - Commit to deployment repo (Acquia, Pantheon model)
 - Integrate with release/deployment tools (Capistrano, Shiplt)



Configuring and Extending



Configuring

- Configurable settings in Gruntconfig.json
 - Source and build paths
 - CLI options for Behat
 - Drush make arguments
 - Theme compilation options
 - Files to include/exclude from package



Example configuration

- Using Drush make `--working-copy` to keep git files

```
1. bash
"projFiles": ["README*", "bin/**"]
},
"phpcs": {
  "path": "vendor/bin/phpcs"
},
"phpmd": {
  "path": "vendor/bin/phpmd"
},
"drush": {
  "cmd": "vendor/bin/drush",
  "make": {
    "args": ["--force-complete", "--working-copy"]
  }
},
"behat": {
  "flags": "--tags ~@wip"
},
"themes": {
  "example_theme": {
    "path": "% config_dir % /themes/example_theme"
```



Extending

- Extensions possible through Gruntfile.js changes
 - Adding new tasks, replacing existing ones
 - Running a shell script as a task
 - Overriding the steps in the default build process



Example extension

- Adding the `grunt-eslint` plugin to perform JS linting

```
1. bash
dc2015$ more Gruntfile.js
module.exports = function(grunt) {

  // Load all plugins and tasks defined by the grunt-drupal-tasks package.
  require('grunt-drupal-tasks')(grunt);

  grunt.loadNpmTasks('grunt-eslint');
  grunt.config('eslint', {
    target: ['build/html/core/misc/drupal.js']
  });

  grunt.task.renameTask('default', 'default-original');
  grunt.registerTask('default', ['default-original', 'eslint']);

};
dc2015$
```



Wrapping Up



Pitching in

- Try using Grunt Drupal Tasks on your next project
- File issues on GitHub to share how it could work better
- Patches welcome! (Submit pull requests on GitHub)
- Take on a “help wanted” issue on GitHub



On the roadmap

- More quality/testing tools (Sass/JS linting, PHPUnit)
- Delegation for Grunt-powered themes
- More options for releasing/deployment
- Build Drupal with Composer
- Integrate with Drupal Console
- More Gadget options (make files, theme support)



Thanks! Questions?

- Evaluate this session, download slides, and find links:

<https://events.drupal.org/node/641>





PHASE2TECHNOLOGY.COM
