BARCELONA

DRUPALCON 2015

BARCELONA
DRUPALCON 2015

Building sites in Drupal 7 with an eye on Drupal 8

Ashraf Abed
Frédéric G. Marand

Session track: Site Building

Introduction

BARCELONA
DRUPALCON 2015

Ashraf Abed

ashrafabed

- Acquia: Officially D8 All In since 07/13/2015
  - Identified as a Leader in new Gartner WCM Magic Quadrant
- Founder of Debug Academy (debugacademy.com)
  - In-person training program: Novice to Drupal Developer!
  - Graduates hired by top companies
  - Located in Washington, DC area
- Acquia Certified Drupal Grand Master
- Upgraded administerusersbyrole module to D8

Follow on twitter: ashabed

Frédéric G. Marand

fgm

- OSInet: performance/architecture consulting for internal teams at larger Drupal accounts
- Core contributor 4.7 to 8.0.x, MongoDB + XMLRPC maintainer + others
- Already 4 D8 customer projects before 8.0.0
- Customer D8 in production since 07/2015
- One production PF6 site using “built for D8” parts

This talk will help you if

- You want to start a Drupal project now/soon
- You are debating whether to ‘wait for D8’ to be widely adopted
- Or you would like to prepare an existing D7 site for migration to D8

What we are explaining

- When to choose D7, D7 +, or D8 today for new projects
- How to save on future upgrades when building your website

Why not just “wait for D8”?

Cost of waiting

- D8 “Ready when it’s ready”
  - Losing the benefits of your new application every month you wait
  - Waiting for an unspecified date
  - IS may even refuse projects based on *.0 products like 8.0.0, hence 8.1.*
- Even when D8 is released, contrib still has to catch up
  - Important contrib (e.g Panels, DS, Rules, Media) underway, just not yet there
  - Lots of contrib won’t make it to d8

Resource availability

- Many project-oriented suppliers mostly not yet D8-ready
- Need to train internal teams, most training suppliers not yet D8-ready

Ability to build “D7+” sites now

- Utilize appropriate site building and contrib module selection strategies
- Write portable code where custom code is needed
- Mimic D8 experience in D7 for users

You can build on D8 now if...

- Your project does not heavily rely on contrib
- You have expert contractors with experience in D8
- You are an expert internal team, especially one with SF2 knowledge in addition to D7
- You are building headless projects, able to deliver most functionality with JS front, which need less from D8 contrib

Start with:
D7, D7+, or D8?

Short *time to market*, short *time to live*

→ D7 with contrib

CC Can Pac Swire, cropped from original work

New simple projects with long TTL

→ **D8**

Little risk, biggest cost savings

CC Mount Pleasant Granary

New complex projects

- Early start time

→ D7 (contrib) + portable code

- Later start time (>6 months), restricted budget

→ D7 (contrib) + portable code

- Later start time, significant budget

→ D8

Provision costs for ongoing maintenance and development of contrib during dev

CC Outi Munter

Complex existing non-Drupal sites to update

- Short Time To Market

→ D7 (contrib) + portable code

- Long Time To Market

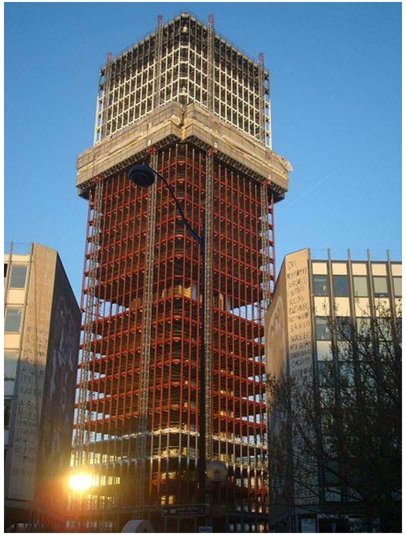
→ D8

Provision costs for ongoing maintenance and development of contrib during dev

- Custom code, continuity of service

→ Prepare for D8

Portable code, cruft cleanup



CC Matthieu Buisson

Including portability to D8

in a D7 site build, or a D6-D7 site upgrade

D8 site building very similar to D6/D7 + Features & friends, just better

Organize your process around a code-driven process

Deploy updates via code, using Git

Features/Strongarm for exporting config

Separate features logically

Module selection: Very low risk

CKeditor

Quick Edit (`quickedit`)

URL, Telephone

RESTful Web Services (`restws`)

Administration Views (`admin_views`)

Entity Form (`entityform`)

- For standard / “relatively light” forms

Workflow

- Due to `workbench_moderation`’s status in D8

Module selection: Medium risk

Drupal Commerce

Rules

Media: cf State of Media

- APIs usable, UI need custom work

Module selection: High risk

Layouts: Panels, Display Suite (`ds`)

Webform

Workbench Moderation

Organic Groups / Domain Access

Front end architecture

Do not use Panels for the sake of simplifying theming

Expect to rebuild the front end during site update

- Renders base theme selection relatively insignificant from the perspective of upgrading

Alternatively, Twig for Drupal (tfd7) brings Twig theming to D7

- *Might* reduce the efforts required to port a theme as-is
- But probably not by much: CSS will still differ a lot

Reduction in contrib availability in D8

Use less contrib, "back to basics" and a little code go a long way

- *Example:* `nodequeue/entityqueue` ⇒ creatively use `entityreference`

Sponsoring port of key contrib modules makes them available to you
⇒ cheaper than custom

How to code for portability

1 Write portable code for big savings on maintenance costs

Write portable code

- Based on past experience with D8 projects:
  - less contrib
  - more custom code because it is simpler to write at high quality level (tests)
- Reusable model + business logic code
- Up to 80 % reusable code
- D8-style code is easier to instrument with tests, hence easier to evolve/maintain/refactor
  - See “Engineering Long-Lasting Software” (Fox, Patterson)

2 Prepare for D8

Prepare

Build a data inventory:

D6/D7 only knows Content vs Configuration (and even then…)

- *Content:* use custom entities and avoid any SQL queries in your code
- *Configuration:* use Variable to be aware of your configuration variables and defaults
- *Cache*: everything D7 has a D8 equivalent, easy to reuse
- *State* is a K/V in D8: backport a state service or use custom tables
- *Settings:* not changed much, just document them, consider 12-factor style
- *Session storage:* not stored the same, but similar uses

Prepare

Think SOLID, write D8-style

- Design your code around a services-based model à la SF2
- Create a minimal “core” service to replace the most-used core functions like `t()`
- Be relentless in pursuing Inversion of Control :
  - Inject whatever the actions need to use to the service objects
  - All functional equivalents and Service Locator-type methods are only for factories and service managers

Prepare

All code except hooks go into PSR-4 classes/interfaces/traits

- Separate Controllers, Forms and Blocks to their own classes
- More generally, design code around decoupled components taking services, and modules as wrappers for them
  - Inspiration: look at Commerce 2, bojanz' session
- Module code is just for hook implementations, use them as Adapters to your Model services:
  - either use a Service Manager to get the service and use the service methods
  - or use the Service Container module, same purpose

3 Apply D8 / PHP standards

Apply standards

- *Autoload:* namespaced PSR-4, not non-namespaced Registry
- *Logging:* use a PSR/3 logger, not `watchdog[_exception]()`
- *HTTP:* use PSR-7/Guzzle 6, not `drupal_http_request()`
- *Testing:* use PHPunit, not Simpletest
- Get familiar with the D8 Caching API: contexts and tags, D7 `DrupalCacheArray`

4 Content Rendering

Content rendering

- When building *render arrays*, prepare cache metadata from the start to ease the D8 port. D7 will just ignore them, unless you can use the render cache API.
- *Custom theme hooks:* favor vector operations over single item operations (perf)
- Organize your *JS/CSS* in libraries, use `#attached`, not `drupal_add_(js|css)()`
- *Markup:*
    - You probably do not care. When the time comes to upgrade to D8, a new design will likely be required anyway.
    - If you care, design your CSS using SMACSS classification / BEM naming principles. It can still help for JS widgets/plugins.

Preparing an existing D7 site for upgrade

Implement best coding practices based on previous slides

Remove instances of unused config

Views and displays

Content types

Fields

Taxonomy vocabularies and terms

Organic Groups

Themes

Features

- Feature overrides

Modules

Rules

User roles

Remediate any instances of core or contrib being directly modified

- Ensure only documented patches are used

Content inventory

- Similar to building a data inventory, but using existing content

Beyond code

8.0.0 and 8.0.* are not the ultimate horizon:

D8 has 8.*.* twice-yearly releases with new features

Sites should no longer be considered as complete for years with just maintenance,

but evolving products delivering new features periodically, taking advantage of the new features in the CMS

BARCELONA
DRUPALCON 2015

Sprint: Friday

Sprint with the Community on Friday.

We have tasks for every skillset.

Mentors are available for new contributors.

An optional Friday morning workshop for first-time sprinters will help you get set up.

Follow @drupalmentoring.

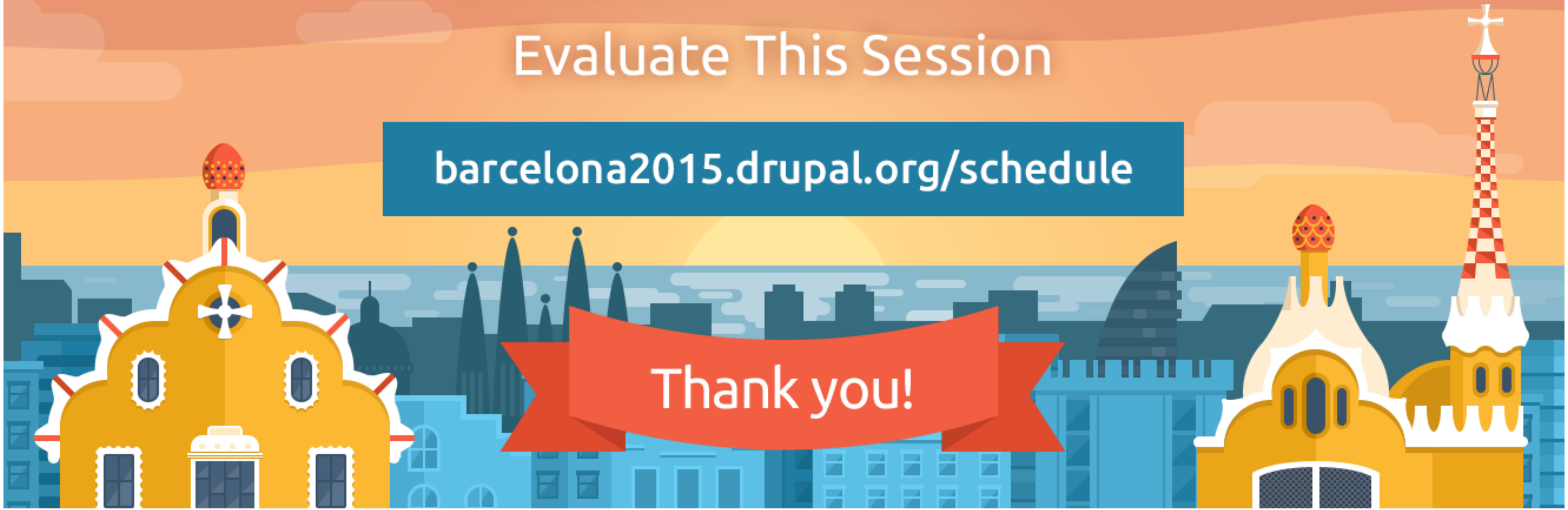
https://www.flickr.com/photos/amazeelabs/9965814443/in/faves-38914559@N03/

BARCELONA
DRUPALCON 2015

What Did You Think?

Evaluate This Session

barcelona2015.drupal.org/schedule



Thank you!

BARCELONA

DRUPALCON 2015