Evaluating Base Themes

Presentation at DrupalCon Munich 2012 by Emma Jane Hogbin.

Slide deck is also available from Slideshare (slideshare.net/emmajane) and SpeakerDeck (speakerdeck.com/emmajane).

About Emma

Emma's first book, *Front End Drupal*, is recognized in the industry as the most important book for Drupal designers. Her second book,*Drupal User's Guide* is quickly becoming the industry standard for introducing new site builders to Drupal. She has been teaching internet technologies since 2002 and has been building Web sites for over ten years. She is a frequent speaker at technical conferences in Canada, the US and Europe and she also likes single malt whiskey. These two things are probably not related.
Emma is a highly sought-after speaker and trainer. Her presentations have taken her to France, Belgium, Hungary, Canada, New Zealand, England and the United States. In addition to her engaging conference presentations, Emma has also worked as a technical college instructor at Humber College and Seneca College, and has worked on curriculum development for Humber College and the Ubuntu Linux distribution.
I'd love to hear from you. Please keep in touch!

@emmajanedotnet | emma@designtotheme.com | http://drupal.org/user/1773

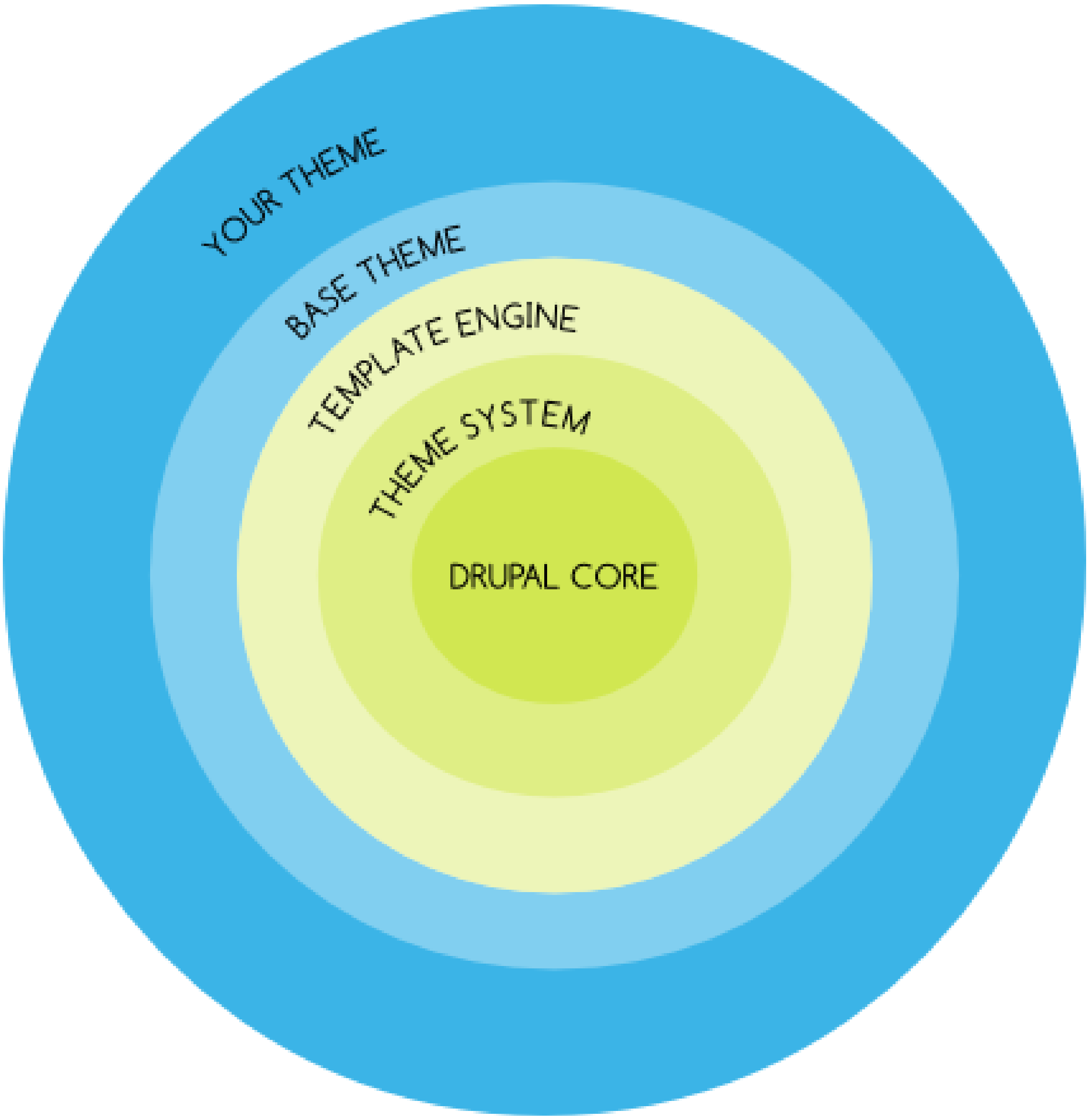
About this presentation

Base themes are misunderstood. Too often in the IRC channel #drupal-themes I watch people asking “Should I use Zen or Omega?” without context of where they’re at in their Drupal learning curve, and more importantly, what they’re trying to accomplish by using a base theme.

Let’s start at the beginning. A base is (according to Google) the part that we use as a foundation; it is the point from which something can develop. A theme is an idea that recurs or pervades a work of art or literature. It is the setting or the ambience given to a leisure venue or activity.

And yet when we combine these two words into a “base theme” we too often forget that a base theme is meant to be a *foundation* on which we build. Base themes are a starting point. They are not merely a stripped down version of a decorated theme. They provide us with value, and have nothing in them that we would want to take away. Indeed, when something integral is missing it is easy for us to add it to our sub-theme.

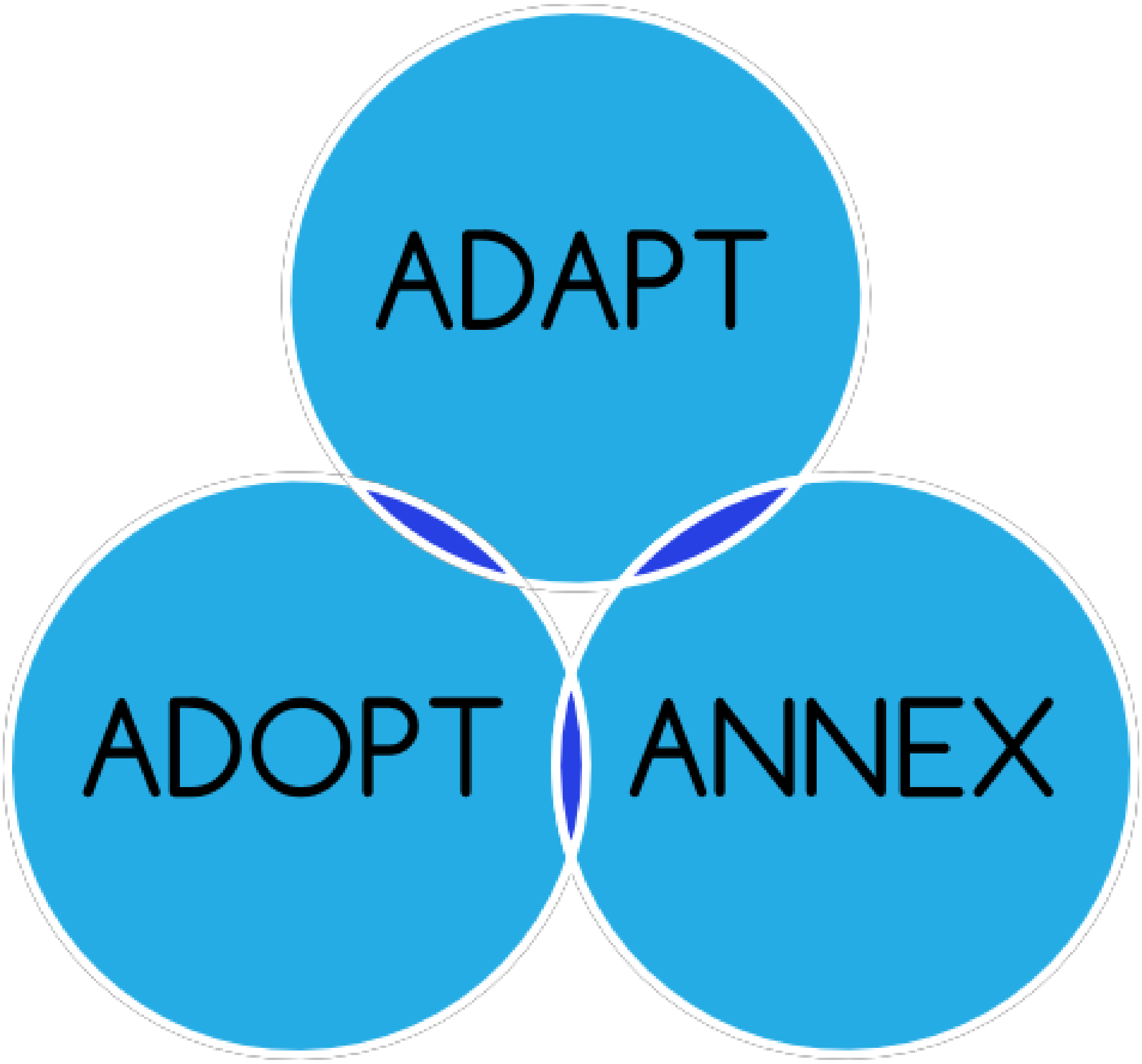
Drupal theming is a series of overrides.



Drupal core, the theme system and the template engine provide us with the infrastructure we need to *override* any markup at the theme layer. The theme, in Drupal, is all-powerful. It has the final "say" on the markup and CSS that is rendered in the browser. Because we have over 8,000 modules, it is common for themers to be working independently of module developers. It is common for themers to need a different version of the markup that is provided by default. The base theme we use, if you use one at all, is all the things you typically need to reset to be able to build your site. It may contain a CSS grid framework, some modified HTML tags for HTML5, a meta tag or two for responsive resets…choosing your base theme wisely for any given project helps you complete that project faster.

There are 3 different kinds of base themes.

Some base themes, of course, fall a little in between the three different kinds, but for the most part, a base theme will reside *mostly* in one of three categories.



1. Adapt Drupal core.

Theme developers strive to improve Drupal core's markup and template variables for improved:

- work flow.
- semantic purity.
- standards-compliance.

Includes: Zen, Boron, Mothership.

Adapting is like renovating a house.

2. Adopt non-Drupal frameworks into Drupal.

Adopt base themes "adopt" an industry best practice for use within Drupal. Typically used for grid frameworks.

Includes: NineSixty, Square Grid, Blueprint.

Adopting is like buying a house.

3. Annex components into a new mega system.

Annex base themes "do all the things".

- Extends Drupal core with additional modules.
- Adapts Drupal core markup (e.g. adds HTML5).
- Adopts a CSS grid-framework.

Includes: AdaptiveTheme, Fusion, Omega.

The Annex is a neighbourhood in Toronto.

Other work flows exist too.

This talk is about base themes. I am ignoring work flows that do not use a base theme used, or use a corporate "starter" kit which is sub-themed by individual departments.

These differences tell us interesting things.

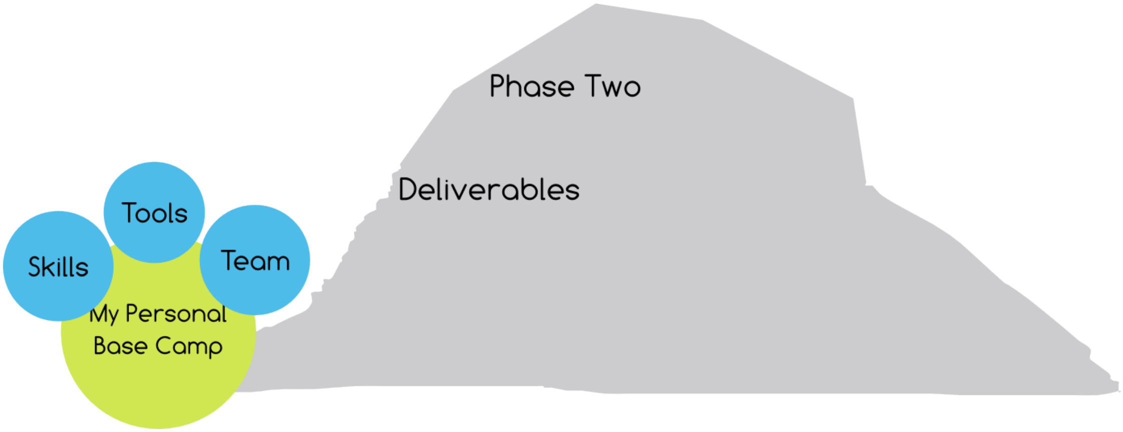
Core is flexible. By allowing front end developers to pull in any grid framework, we future proof ourselves against whatever the next great trend is on the web. No one saw responsive coming, thank goodness we didn't "hard code" the 960 CSS Grid Framework into core.

Core is broken. The annex themes show us how we've outgrown "themes". Designers have finally prevailed and we can no longer simply slap some CSS on Drupal to "theme" it. Our markup still needs work. Thank goodness theme developers care.

What about me? Where do I start?

This presentation is about helping you choose the right base theme for your needs. Before we can proceed, I need to know a little more about where you are, and where you want to go. You don't get a sherpa if you're only going up one flight of stairs!

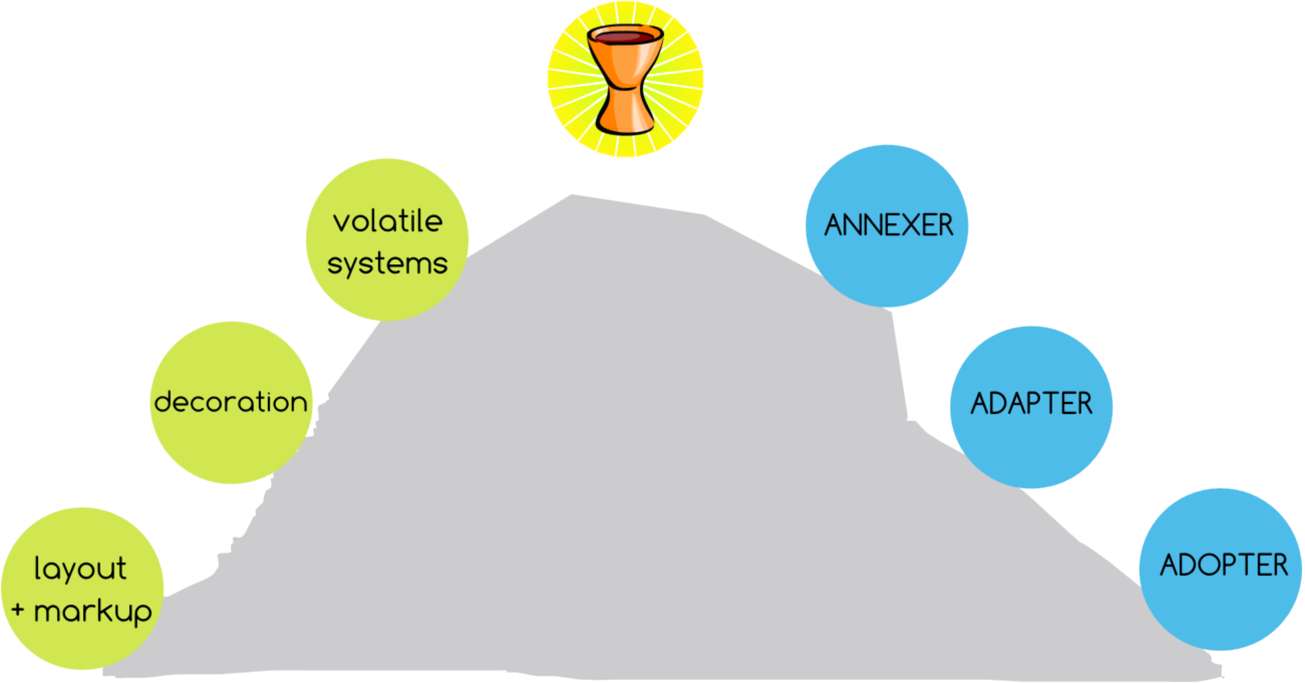
Define your personal base camp. *What are your skills?* HTML, CSS, JS, PHP, etc



Your tools? SASS. *Your team?* Do you work alone, or with a team of developers? Is your client tech-savvy? Will they want to manipulate their site after you hand it over to them? *What are your deliverables?* What are you contractually obligated to do? Does your client care about HTML5? Seriously, now. How important is source order and SEO? *What is "phase two" for this site?* How will the client use your theme?

When I teach theming I tend to get a lot of people who are new to front end development. I take them through the following progression:

1. **layout + markup**. Typically I use an "adopter" base theme, such as NineSixty. http://drupal.org/project/ninesixty + http://drupal.org/project/domicile.
2. **decoration**. Next we focus on CSS enhancements and start working more in-depth with Drupal's theme system. At this point I tend to switch to an "adapter" base theme, such as Zen. http://drupal.org/project/zen.
3. **volatile systems**. With a solid foundation, and practice with phptemplate, markup and CSS, I finally move my students into an "annexer" base theme. Now they are learning an entirely new suite of tools. Typically they don't need to do much PHP work, and can focus on CSS enhancements, and learning the new layout tools. I have public workbooks on both Fusion (http://drupal.org/project/fusion) and Omega (http://drupal.org/project/omega) available at designtotheme.com. My own thinking has shifted though, and I tend to point people to AdaptiveTheme (http://drupal.org/project/adaptivetheme) as a good first "annexer" base theme.



What can I expect from a base theme?

Recap: your average theme typically has the following types of files:

- info file to control which external files are used
- CSS files
- PHPtemplate files (tpl.php) for markup
- Javascript and jQuery files, but typically not libraries
- phptemplate.php file which includes new variables for PHPtemplate files

Base themes are meant to be the *foundation* that you use to build out your design. They will never give you “decoration” such as colours and fonts. They will also *typically* not be used to configure the layout of content (or blocks) within regions on your site as we expect this to change from site to site.

Base themes ADAPTING Drupal introduce fewer theme-specific “isms”.

In the theme folder, expect to find:

- tpl.php files with (p)resets for markup.
- CSS files with useful selectors.
- a template.php file with minor enhancements and new variables.

Base themes ADOPTING outside technologies tend to be CSS-focused in their enhancements.

In the theme folder, expect to find:

- CSS grid framework files.
- tpl.php files with non-Drupal classes that match the grid framework.
- template.php file with framework-related math for conditional regions.

Base themes ANNEXING other systems will include all the other stuff + its own glue.

In the theme folder, expect to find:

- All the stuff the ADOPTER base theme had.
- README.txt file with extra steps on how to install and configure all components.
- module dependencies
- GUI for layout
- A lot of extra code with theme-specific logic.

NB: Don’t expect the tpl.php files to have “normal” HTML layout.

There is no fixed criteria to evaluate base themes.

Make your own chart that lists the criteria important to you. Keep an eye out for Michael’s article in Drupal WatchDog (http://drupalwatchdog.com/issue/toc/2/1). Your chart may include:

- # of installs
- lines of comments vs. lines of code
- buzzword compliance
- in-code documentation
- ease of installation
- UI help text
- online documentation
- support (community + professional)

Base Themes According to Their Developers

While preparing for this presentation, Morten (http://morten.dk/) gave me an excellent idea. He suggested I ask the base theme developers to evaluate their own systems. This is the email I sent out:

> At DrupalCon Munich I'm giving a talk on base themes.

> As part of the presentation I'd like to highlight a few base themes. I'd like to include an honest evaluation of what the *developers* of the base theme think of their own theme's strengths and weaknesses. I'd love it if you'd take part!
>
> Please email me back the answers to the following questions. Each answer should be not more than 140 characters. I will trim *at* 140 characters. ;)
>
> 1. Describe your theme in a nutshell.
> 2. Describe its best feature.
> 3. What is the worst thing about your base theme?
> 4. What notable difference is there from other themes?

I contacted the following base themes. Either via their personal email address if I knew them personally, or through their drupal.org contact form. The person whose name was listed as "owning" the project was contacted. All responses received have been included in this presentation.

- Mothership
- Zen
- NineSixty
- Blueprint
- Square Grid Theme
- Fusion
- Omega
- AdaptiveTheme

drupal.org/project/zen

1. Zen is a Sass/Compass-powered, SMACSS-lovin', drush-buildable HTML5 Drupal starter theme with a responsive, mobile-first grid design.
2. All of Zen's components are concerned with performance: lean HTML5 markup, SMACSS-style CSS rules and organization, and light-weight PHP.
3. Its total lack of omniscience. If you don't know CSS well, Zen won't be able to build your sub-theme for you.
4. Zen gives you the Sass tools to easily build lean, custom designs with any responsive

layout imaginable, not a giant configuration form.

drupal.org/project/mothership

1. Cleans up the fucking markup & gimme wtf i want
2. looks like stark & you have to know wtf you doing ;)
3. you get to swear like a pirate & wear little white sailor hats

drupal.org/project/detamo

1. Detamo is a responsive base theme featuring 3 separate grids. Grids are fluid; based on a 960px desktop, a 640px tablet and a 320px Mobile.
2. It's easy use Detamo to make an existing 960-based theme responsive. Just add the grid classes for tablet and mobile to your template.
3. It won't appeal to those who don't like (non-semantic) grid classes. (And you'll have them for the desktop, tablet and mobile grids).
4. The responsive layout is in the template file so you don't need to write css to control the layout.

drupal.org/project/ninesixty

1. NineSixty is a Drupal implementation of the 960 Grid System. It's a CSS framework rather than a true "base theme."
2. It's very lightweight, flexible, and makes no assumptions about the look and feel.
3. With the rise of Sass/Compass and responsive design, the 960 Grid System is no longer the best starting point for grid-based design.
4. As a CSS framework, it can be "stacked" with other frameworks or base themes. Or you can simply copy the grid CSS file int your theme.

drupal.org/project/squaregrid

1. A lightweight base theme leveraging The Square Grid thesquaregrid.com framework for rapid, lightweight theming.
2. Square Grid is lightweight and flexible, and otherwise stays out of your way. A child theme could be done in 2-400 lines of (commented) CSS.
3. Lightweight = few configurable settings. If themer wants to change layouts, comfort w light php in template.php and .tpl files is needed.

4. This is not a theme that tries to make your coffee. Settings are minimal. The power is in the Square Grid css.

drupal.org/project/blueprint

[I love the honesty of this response. I think it tells us a lot about our community's desire to stay up-to-date and relevant.]

Thanks for contacting me about Blueprint. Unfortunately at this point, I will have to back down from this opportunity to answer questions about Blueprint. Since I took over the development, my interests have moved beyond what Blueprint provides and I am no longer able to maintain or develop it. At this time, I would, however, love to find another developer who can take over the theme and really make it shine.

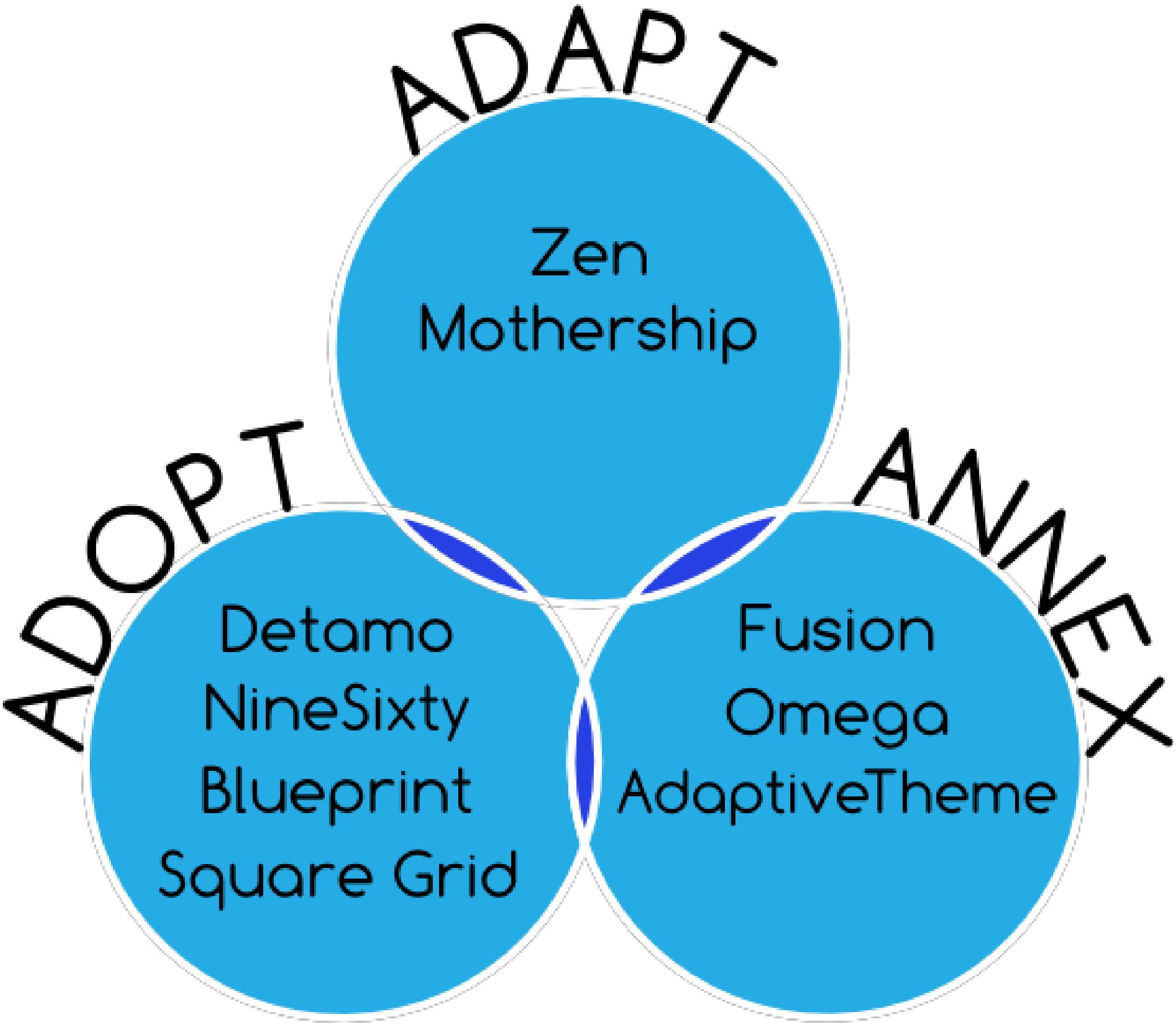
Blueprint is great, but as I have started to us responsive design techniques in my designs, I have moved beyond the limits of blueprint and begun using SCSS.

drupal.org/project/adaptivetheme

1. Responsive, HTML5, uses responsive plugins to support Panels, Panelizer and Display Suite, uses SASS, normalize, focus on accessibility
2. Tools for mobile: Browscap integration, responsive JS, responsive layout, integration with Panels & Display Suite, extensive theme settings
3. Layout plugins more complex than standard Panels plugins, necessity to abstract from Drupal to gain performance, undocumented Easter eggs ;)
4. Pluggable layout system, Responsive JavaScript, Browscap integration (move blocks, unset regions, use $is_mobile to test for mobile context)

drupal.org/project/omega

1. Base theme for developing mobile friendly themes. Currently 3rd highest used theme on Drupal.org
2. Responsive out of the box, adapting to today's and tomorrow's devices.
3. That it fails to make my coffee or clean my house.
4. One of the most passionate communities I've ever seen surround Omega, and support the documentation and improvement of the code base. It's a *[remainder omitted as this was trimmed at 140 characters as promised]*



You promised me a winner!

And the winner is...Mummy lied. No one gets to win at base themes. Seriously.

> "Once the issue is framed,
> if you accept the framing, if you accept the language,
> it's all over."
> George Lakoff.

While preparing this presentation, and reflecting on my years as a Drupal themer, I came to realize it I was asking the wrong question when I asked myself "what is the best base theme"? The right question started to turn in my head: what do base themes teach us about our theming system in Drupal core?

insert. rant. here.

The theming ecosystem in Drupal is a difficult place. There are probably fewer than 200 full time Drupal themers (I'm not one of them). Most also do a bit of site building, or a bit of non-Drupal work, or a bit of Web development, or a bit of ... you get the point. In WordPress land there are thousands of professional themers. Their "base" is relatively stable. As we abstract Drupal (e.g. fields in core) and make it into a product builder, instead of a product, it becomes more and more difficult for the occasional themer to get their job done.

We are looking at "simplifying" themes with Twig, and we know that the *real* security problems lie in the theme layer, where no one keeps a careful watch (you *can* add an SQL query to your theme...you *shouldn't* but you *can*). In D8 we're building a product builder (some call this a 'framework'); but we haven't given themers a front end framework to work with.

Putting a hat on a moose doesn't mean it's suddenly not a moose. Are we really diving deep enough by changing just the templating engine? What would happen if we eliminate the concept of "themes"? What do we really want to do when we "make Drupal look pretty"? or perhaps more accurately these days "make Drupal look like the designer asked".

A "theme" = granular Layout of components + Page layout + Style + Libraries/scripts . Or if we look at some of the premium themes available for WordPress, a theme is an install profile of Modules + Layout + Groups of styles (skinr) + Markup + Libraries/scripts + palette.

But it's not just about the code.

We need to stop treating themers like a second class developer. We need to stop trying to make theming "easy". Have you looked at web technologies recently? The web grew up. It's **complicated** out there. We need to respect "themers" as the real developers they are and teach them how to write modules. Create a front end developer framework that makes it easier to drop in non-Drupal/web best practices. Stop pretending that themes are interchangeable and can just be dropped onto a site. We need to acknowledge that "themes" are more than what you can punt into a theme folder. They are views, view modes, image sizes, fields, markup, javascript libraries, etc.

Let's rethink this whole "theme" thing. Let's start treating themers like the real developers they are Let's examine the solutions base themes are creating and address the underlying problems. Let’s force themers to imagine a new front end tool kit.

The Theming Pledge.

“I promise to help make theming rock in Drupal 8.”

Take the pledge today, and let’s get this rocking in Drupal 8!

The work has already started!

http://drupal.org/community-initiatives/drupal-core

http://drupal.org/node/1089096

“Fix it in 8” Sessions to Attend at DrupalCon.

A new theme layer in D8

Wednesday; 15:45-16:45 in the (Westin) Barcelona room.
http://munich2012.drupal.org/content/new-theme-layer-drupal-8

Designer-friendly theme system in D8

Tuesday; 17:00-18:00 in the (Sheraton) Asam room.
http://munich2012.drupal.org/program/sessions/designer-friendly-theme-system-drupal-8