DRUPALCON SYDNEY

FRONT END TRACK | FEBRUARY 8 2013

THE NEW THEME LAYER IN DRUPAL 8

THE NEW THEME LAYER IN D8

Jen Lampton ~ @jenlampton

Piers Warmers ~ @warmers

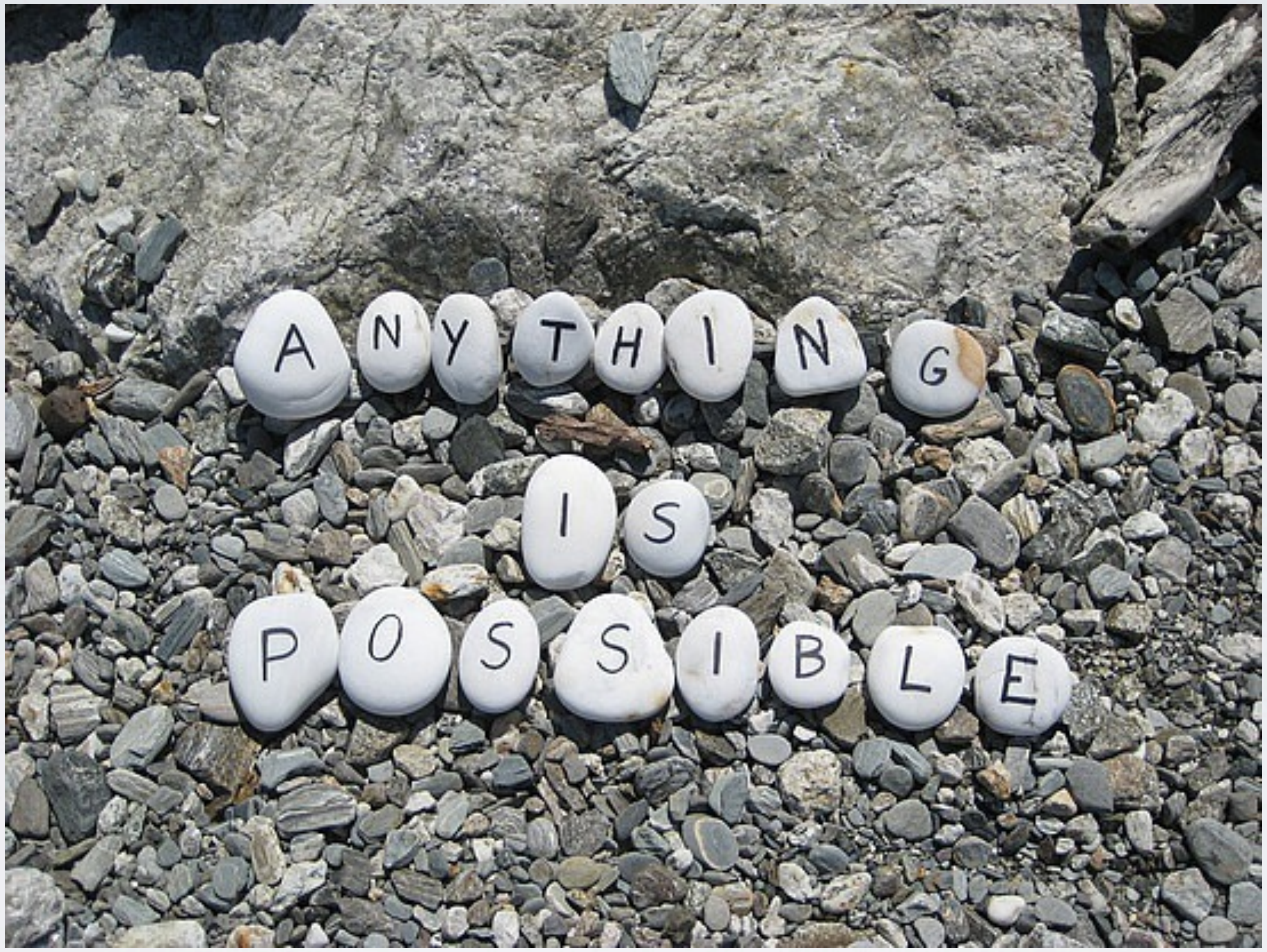
John Albin Wilkins ~ @johnalbin

Morten Birch Heide-Jorgensen ~ @mortendk

WHY?

DRUPAL 7: POSITIVE

DRUPAL 7: POSITIVE

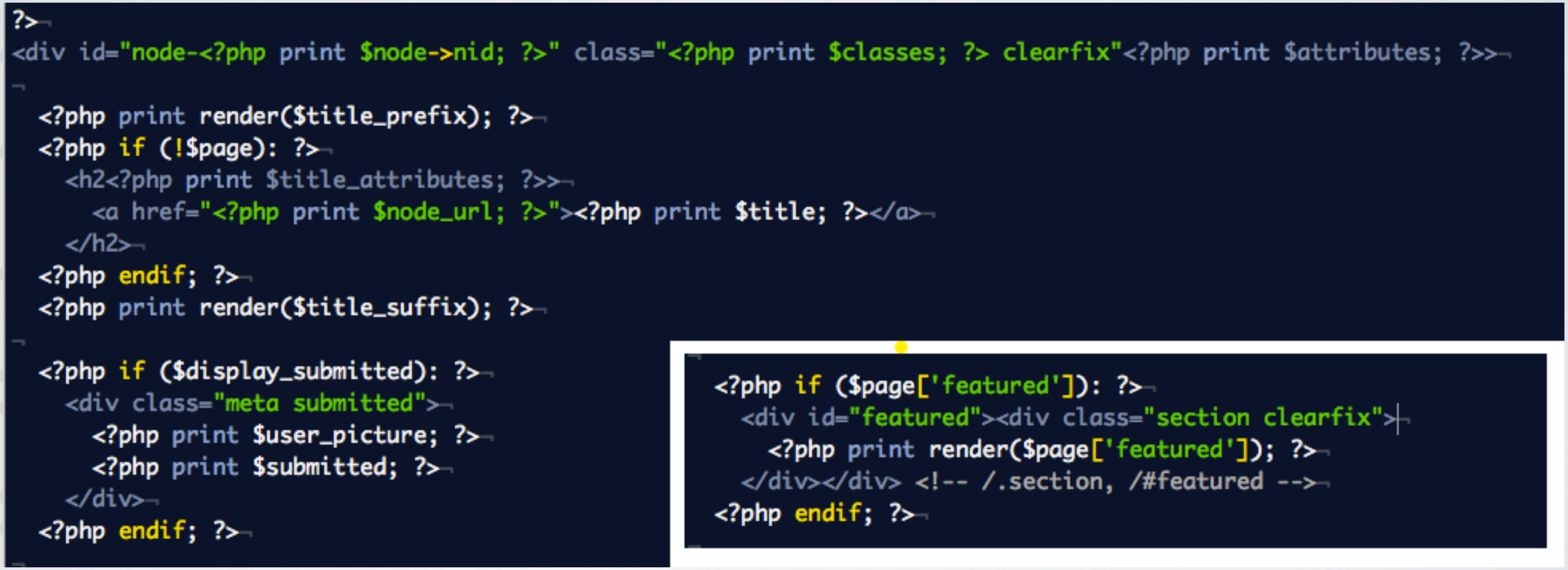


anything is possible

DRUPAL 7: NEGATIVE

DRUPAL 7: NEGATIVE

- Drupal-specific template conventions



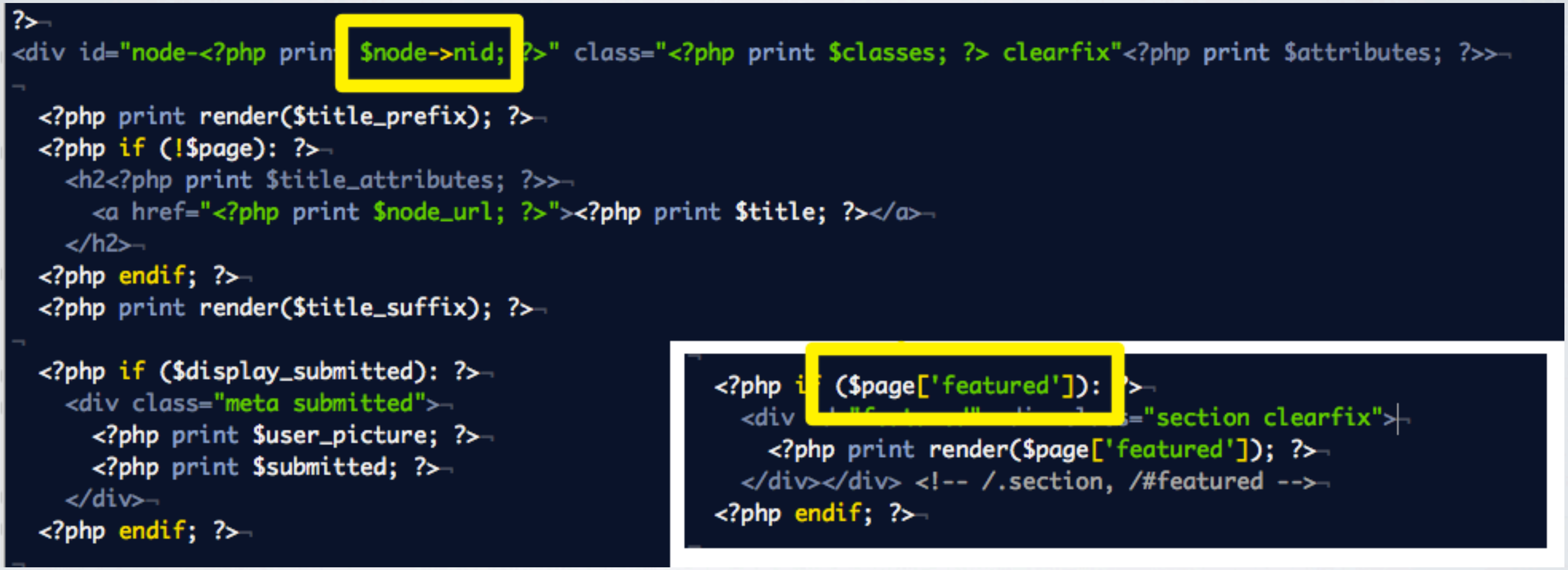
```
?>
<div id="node-<?php print $node->nid; ?>" class="<?php print $classes; ?> clearfix"<?php print $attributes; ?>>

  <?php print render($title_prefix); ?>
  <?php if (!$page): ?>
    <h2<?php print $title_attributes; ?>>
      <a href="<?php print $node_url; ?>"><?php print $title; ?></a>
    </h2>
  <?php endif; ?>
  <?php print render($title_suffix); ?>

  <?php if ($display_submitted): ?>
    <div class="meta submitted">
      <?php print $user_picture; ?>
      <?php print $submitted; ?>
    </div>
  <?php endif; ?>
```

```
<?php if ($page['featured']): ?>
  <div id="featured"><div class="section clearfix">
    <?php print render($page['featured']); ?>
  </div></div> <!-- /.section, /#featured -->
<?php endif; ?>
```

Drupalism *noun* Something that only exists in Drupal.

DRUPAL 7: NEGATIVE

- Mixed data types (strings, objects & arrays)



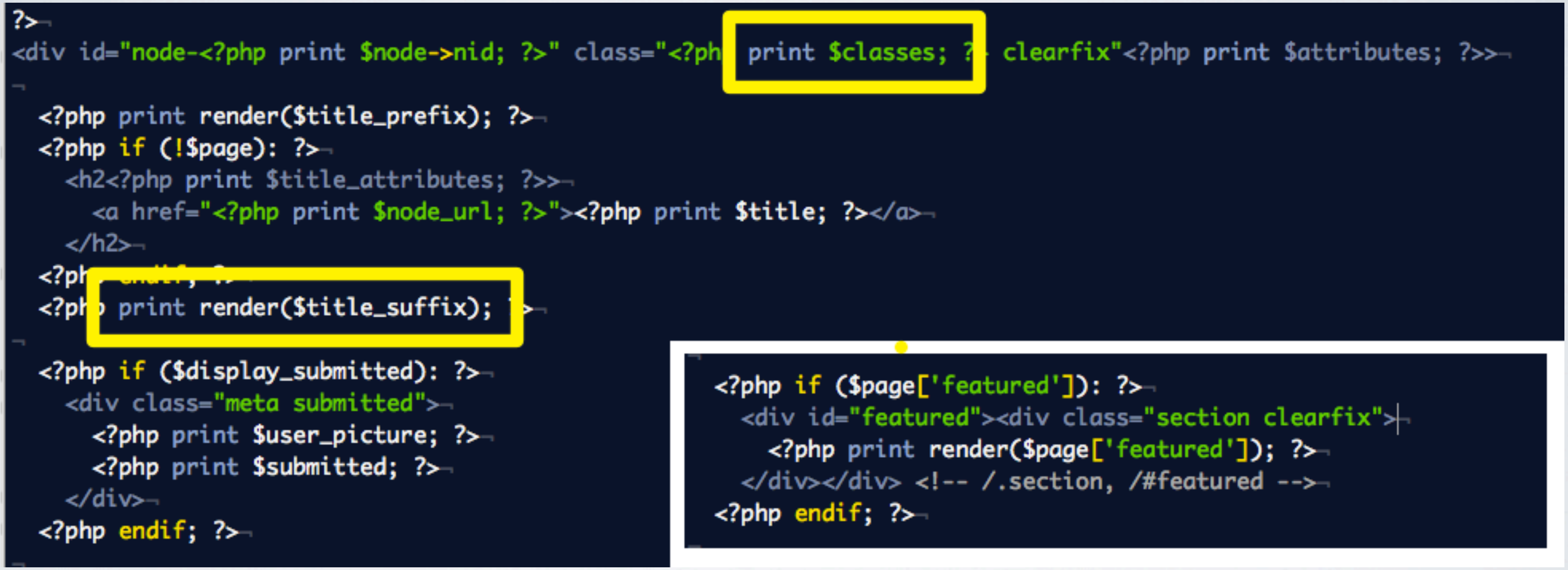
```
?>
<div id="node-<?php print $node->nid; ?>" class="<?php print $classes; ?> clearfix"<?php print $attributes; ?>>

  <?php print render($title_prefix); ?>
  <?php if (!$page): ?>
    <h2<?php print $title_attributes; ?>>
      <a href="<?php print $node_url; ?>"><?php print $title; ?></a>
    </h2>
  <?php endif; ?>
  <?php print render($title_suffix); ?>

  <?php if ($display_submitted): ?>
    <div class="meta submitted">
      <?php print $user_picture; ?>
      <?php print $submitted; ?>
    </div>
  <?php endif; ?>
```

```
<?php if ($page['featured']): ?>
  <div id="featured"><div class="section clearfix">
    <?php print render($page['featured']); ?>
  </div></div> <!-- /.section, /#featured -->
<?php endif; ?>
```

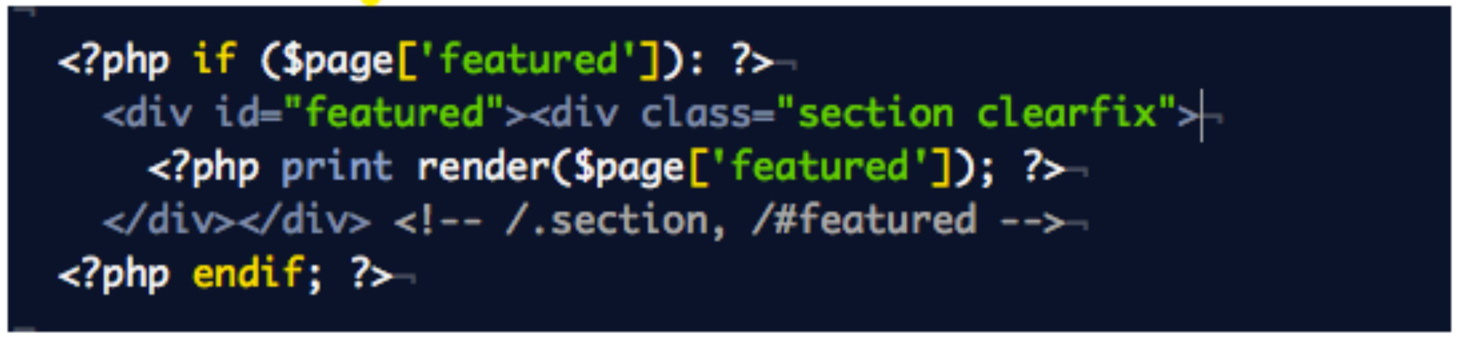
Object or Array?

DRUPAL 7: NEGATIVE

- Different methods of printing



```
?>
<div id="node-<?php print $node->nid; ?>" class="<?ph print $classes; ? clearfix"<?php print $attributes; ?>>

  <?php print render($title_prefix); ?>
  <?php if (!$page): ?>
    <h2<?php print $title_attributes; ?>>
      <a href="<?php print $node_url; ?>"><?php print $title; ?></a>
    </h2>
  <?ph  endif; ?>
  <?ph  print render($title_suffix);  >

  <?php if ($display_submitted): ?>
    <div class="meta submitted">
      <?php print $user_picture; ?>
      <?php print $submitted; ?>
    </div>
  <?php endif; ?>
```

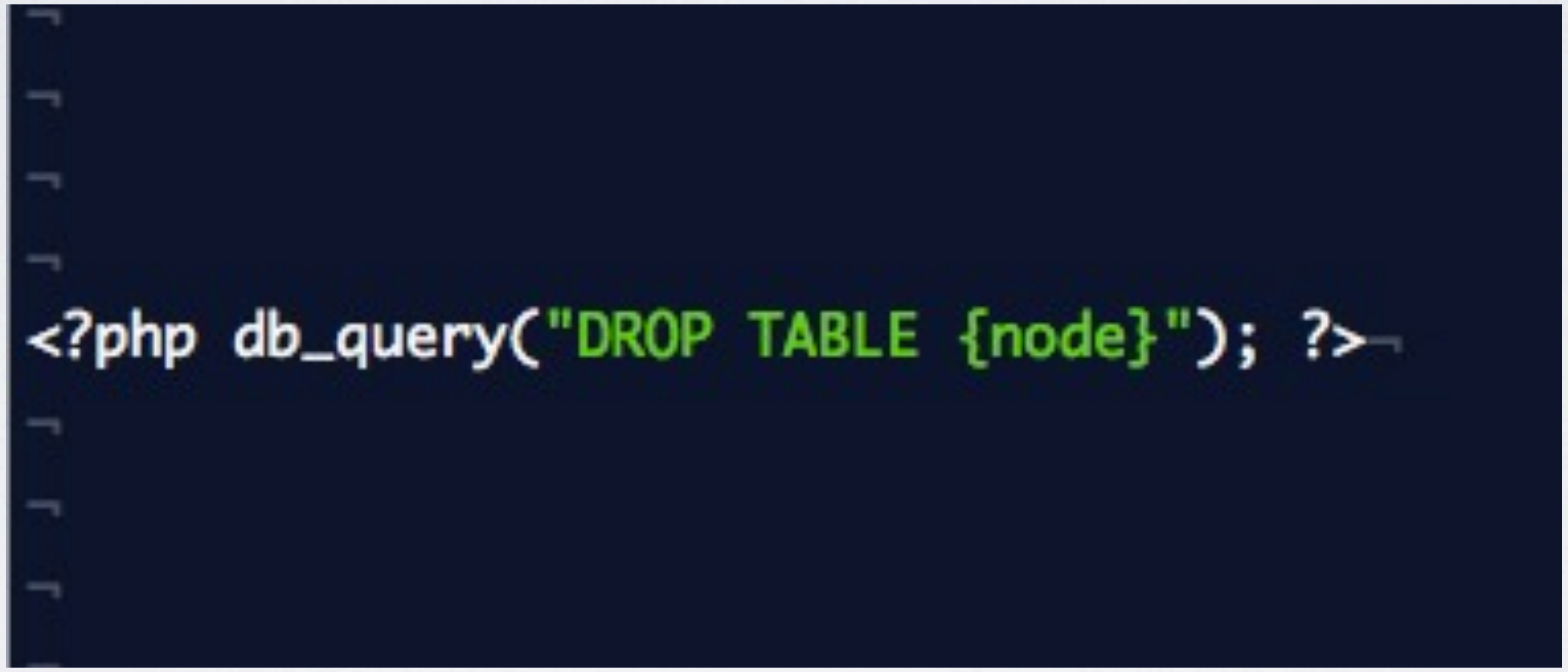


```
<?php if ($page['featured']): ?>
  <div id="featured"><div class="section clearfix">
    <?php print render($page['featured']); ?>
  </div></div> <!-- /.section, /#featured -->
<?php endif; ?>
```

print or print render() ?

DRUPAL 7: NEGATIVE

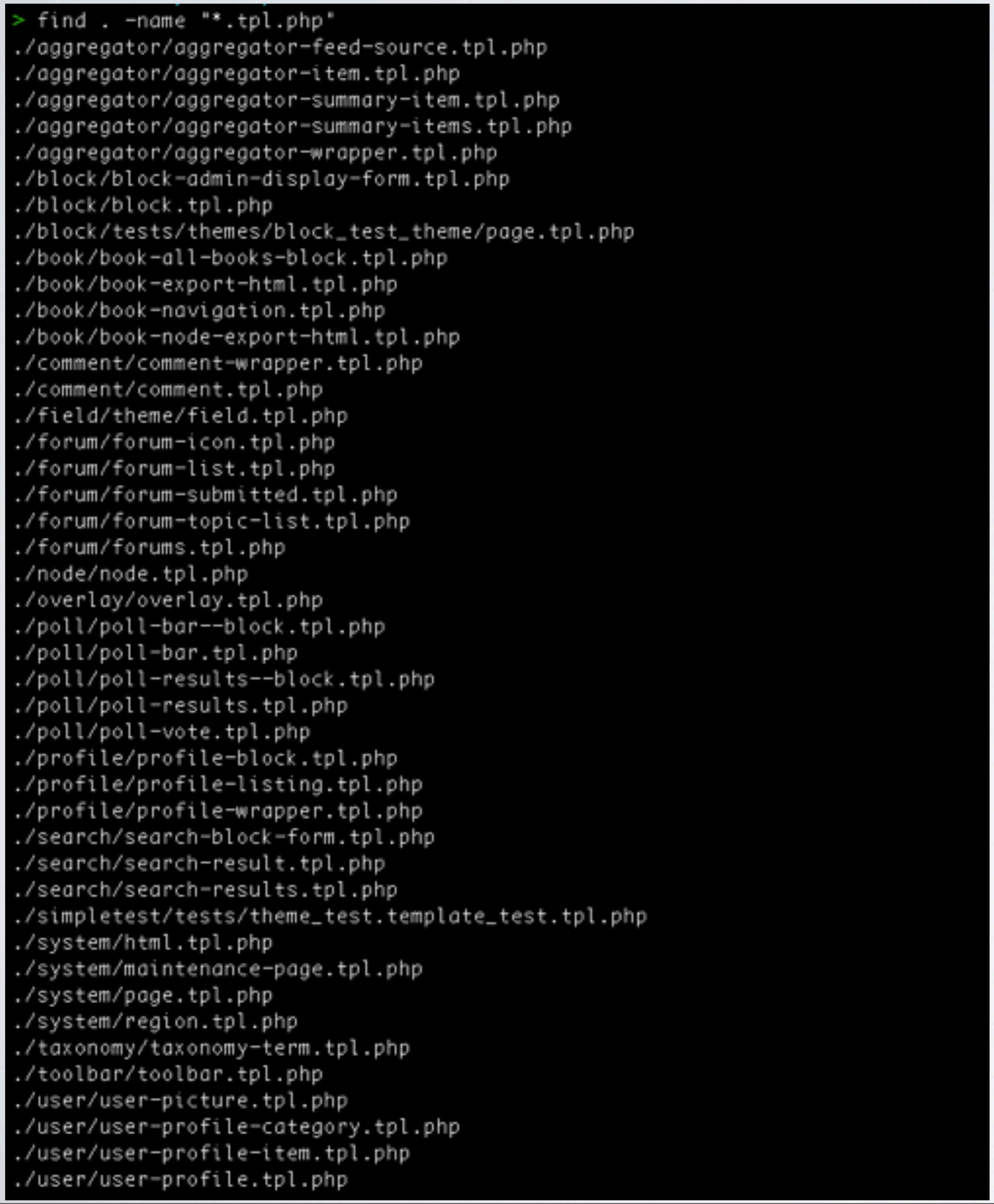
- PHPTemplate is insecure



```
<?php db_query("DROP TABLE {node}"); ?>
```

PHP is insecure

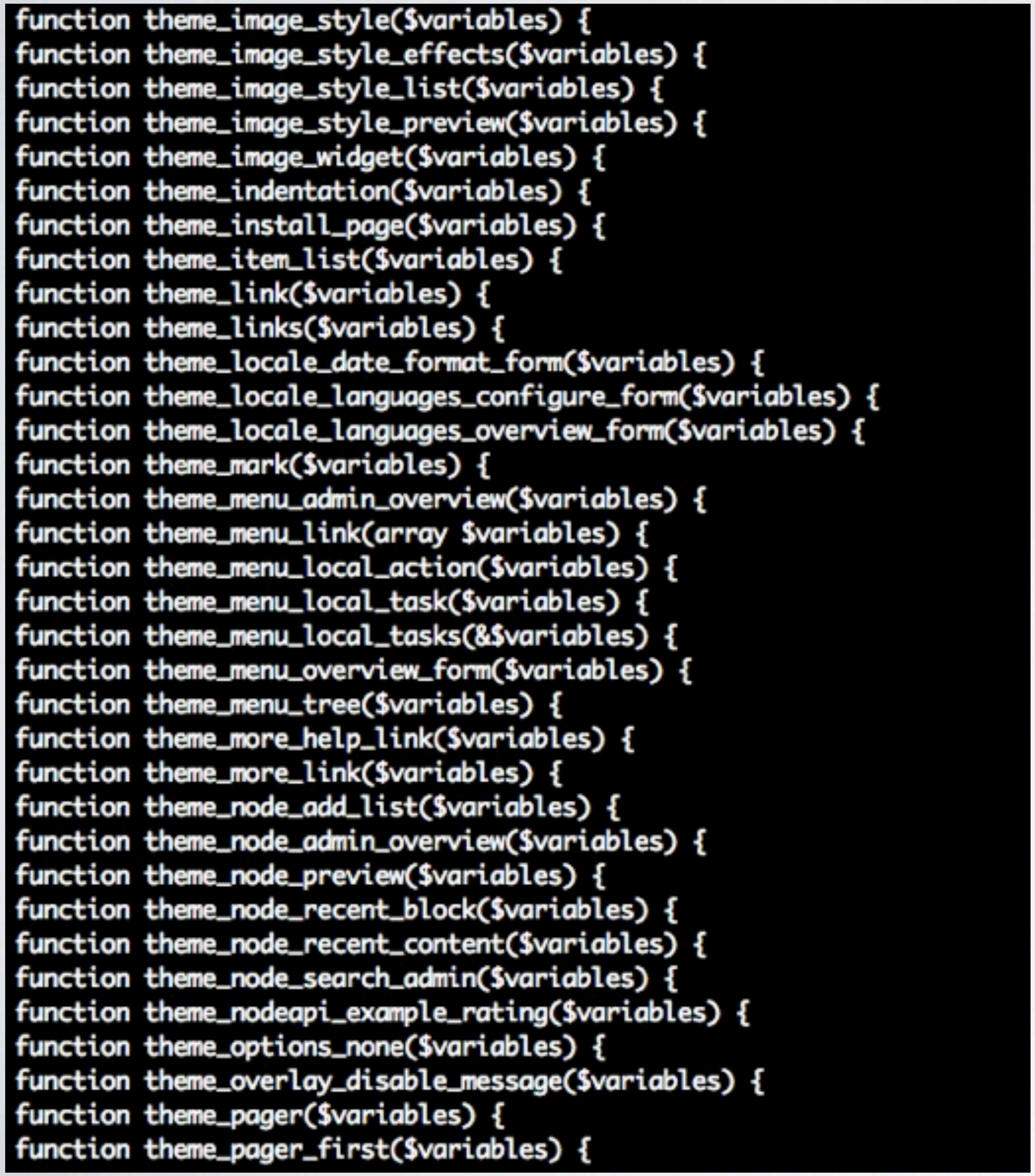
DRUPAL 7: NEGATIVE



```
> find . -name "*.tpl.php"
./aggregator/aggregator-feed-source.tpl.php
./aggregator/aggregator-item.tpl.php
./aggregator/aggregator-summary-item.tpl.php
./aggregator/aggregator-summary-items.tpl.php
./aggregator/aggregator-wrapper.tpl.php
./block/block-admin-display-form.tpl.php
./block/block.tpl.php
./block/tests/themes/block_test_theme/page.tpl.php
./book/book-all-books-block.tpl.php
./book/book-export-html.tpl.php
./book/book-navigation.tpl.php
./book/book-node-export-html.tpl.php
./comment/comment-wrapper.tpl.php
./comment/comment.tpl.php
./field/theme/field.tpl.php
./forum/forum-icon.tpl.php
./forum/forum-list.tpl.php
./forum/forum-submitted.tpl.php
./forum/forum-topic-list.tpl.php
./forum/forums.tpl.php
./node/node.tpl.php
./overlay/overlay.tpl.php
./poll/poll-bar--block.tpl.php
./poll/poll-bar.tpl.php
./poll/poll-results--block.tpl.php
./poll/poll-results.tpl.php
./poll/poll-vote.tpl.php
./profile/profile-block.tpl.php
./profile/profile-listing.tpl.php
./profile/profile-wrapper.tpl.php
./search/search-block-form.tpl.php
./search/search-result.tpl.php
./search/search-results.tpl.php
./simpletest/tests/theme_test.template_test.tpl.php
./system/html.tpl.php
./system/maintenance-page.tpl.php
./system/page.tpl.php
./system/region.tpl.php
./taxonomy/taxonomy-term.tpl.php
./toolbar/toolbar.tpl.php
./user/user-picture.tpl.php
./user/user-profile-category.tpl.php
./user/user-profile-item.tpl.php
./user/user-profile.tpl.php
```

Too many template files

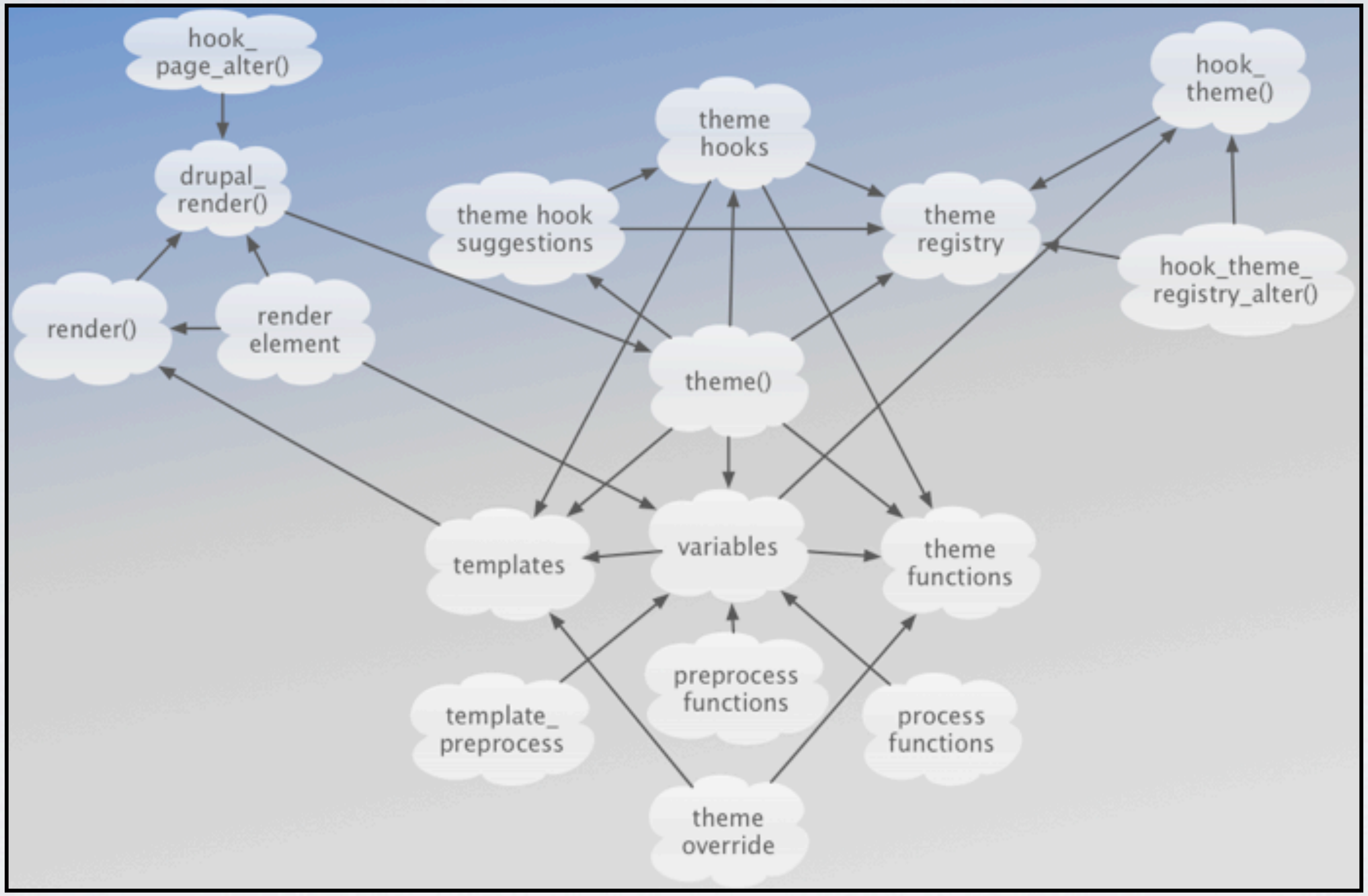
DRUPAL 7: NEGATIVE



```
function theme_image_style($variables) {
function theme_image_style_effects($variables) {
function theme_image_style_list($variables) {
function theme_image_style_preview($variables) {
function theme_image_widget($variables) {
function theme_indentation($variables) {
function theme_install_page($variables) {
function theme_item_list($variables) {
function theme_link($variables) {
function theme_links($variables) {
function theme_locale_date_format_form($variables) {
function theme_locale_languages_configure_form($variables) {
function theme_locale_languages_overview_form($variables) {
function theme_mark($variables) {
function theme_menu_admin_overview($variables) {
function theme_menu_link(array $variables) {
function theme_menu_local_action($variables) {
function theme_menu_local_task($variables) {
function theme_menu_local_tasks(&$variables) {
function theme_menu_overview_form($variables) {
function theme_menu_tree($variables) {
function theme_more_help_link($variables) {
function theme_more_link($variables) {
function theme_node_add_list($variables) {
function theme_node_admin_overview($variables) {
function theme_node_preview($variables) {
function theme_node_recent_block($variables) {
function theme_node_recent_content($variables) {
function theme_node_search_admin($variables) {
function theme_nodeapi_example_rating($variables) {
function theme_options_none($variables) {
function theme_overlay_disable_message($variables) {
function theme_pager($variables) {
function theme_pager_first($variables) {
```

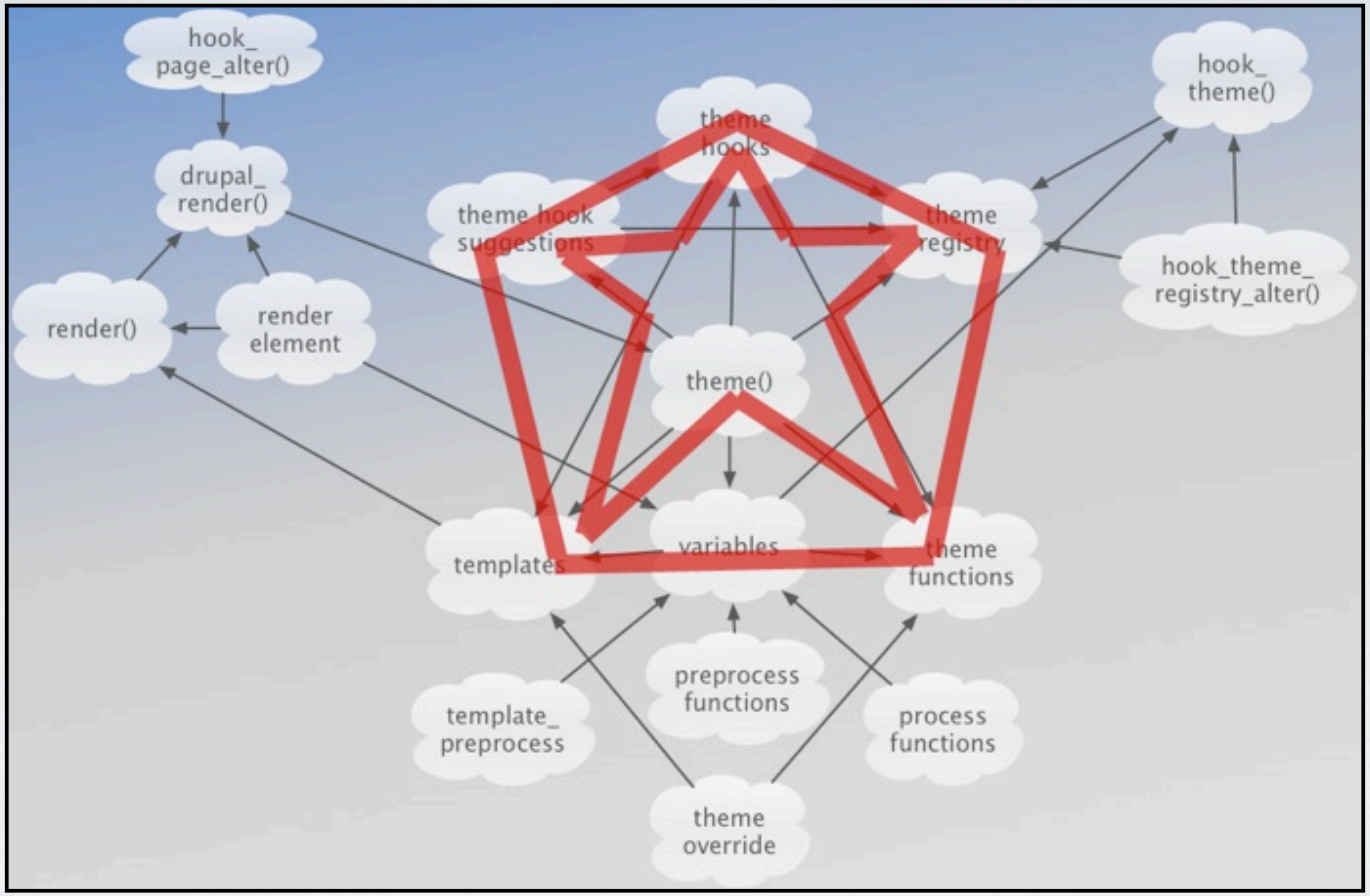
Too many theme functions

DRUPAL 7: NEGATIVE



too many complex of subsystems

DRUPAL 7: NEGATIVE



Satanic?

DRUPAL 7: NEGATIVE

Drupal 7 is too hard to learn!

THIS IS A LOT OF STUFF

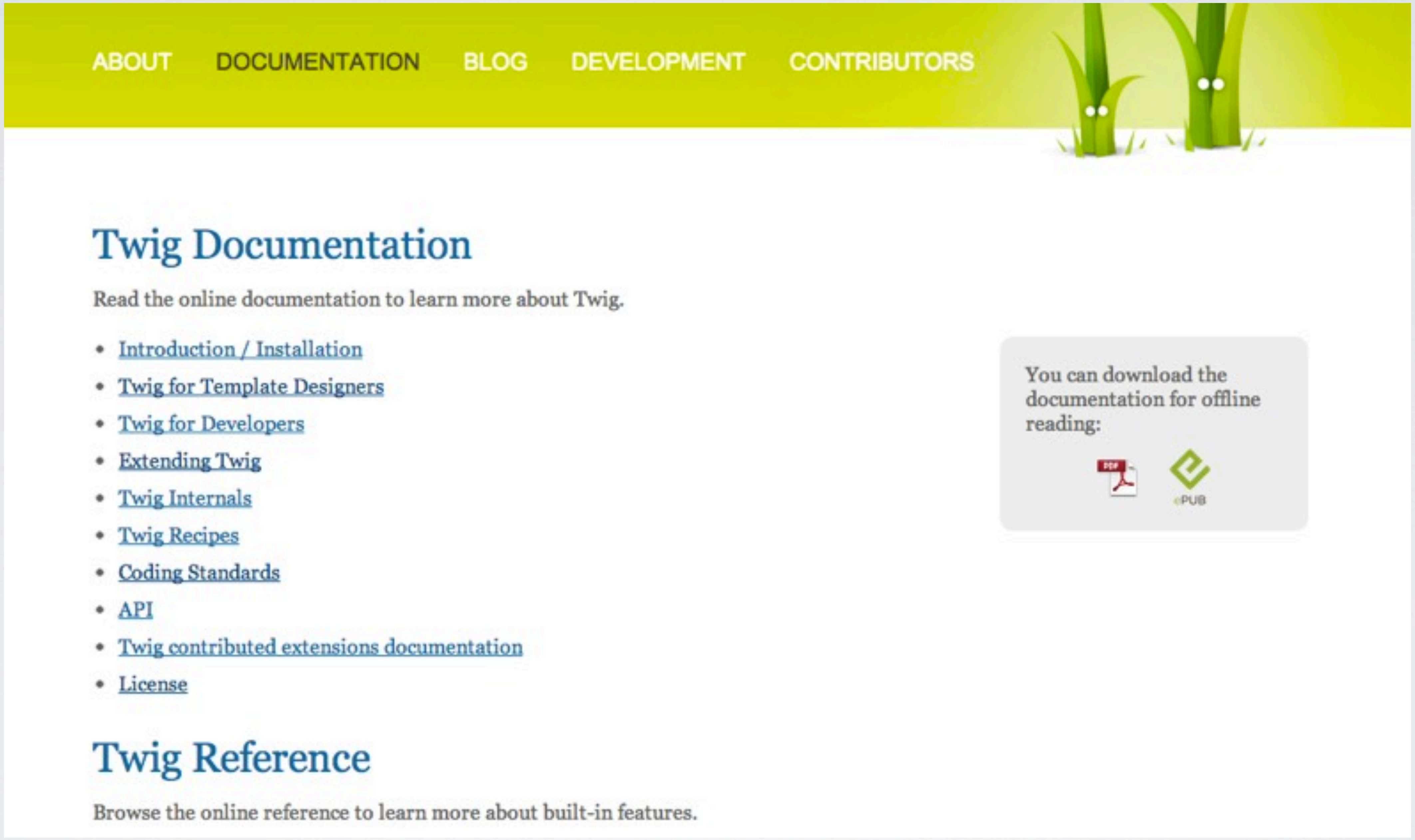
- Drupal-specific template conventions
- Mixed data types (strings, objects & arrays)
  $node->nid vs $content['links']
- Different methods of printing
  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure
- Two ways to override markup
  Templates: *.tpl.php vs functions: theme_foo()
- Many template files, many similar theme functions
- Complex mix of subsystems
- Difficult to learn for newcomers (and D6 veterans)

DRUPAL 8

DRUPAL 8

TWIG

TWIG



Twig Documentation

Read the online documentation to learn more about Twig.

- Introduction / Installation
- Twig for Template Designers
- Twig for Developers
- Extending Twig
- Twig Internals
- Twig Recipes
- Coding Standards
- API
- Twig contributed extensions documentation
- License

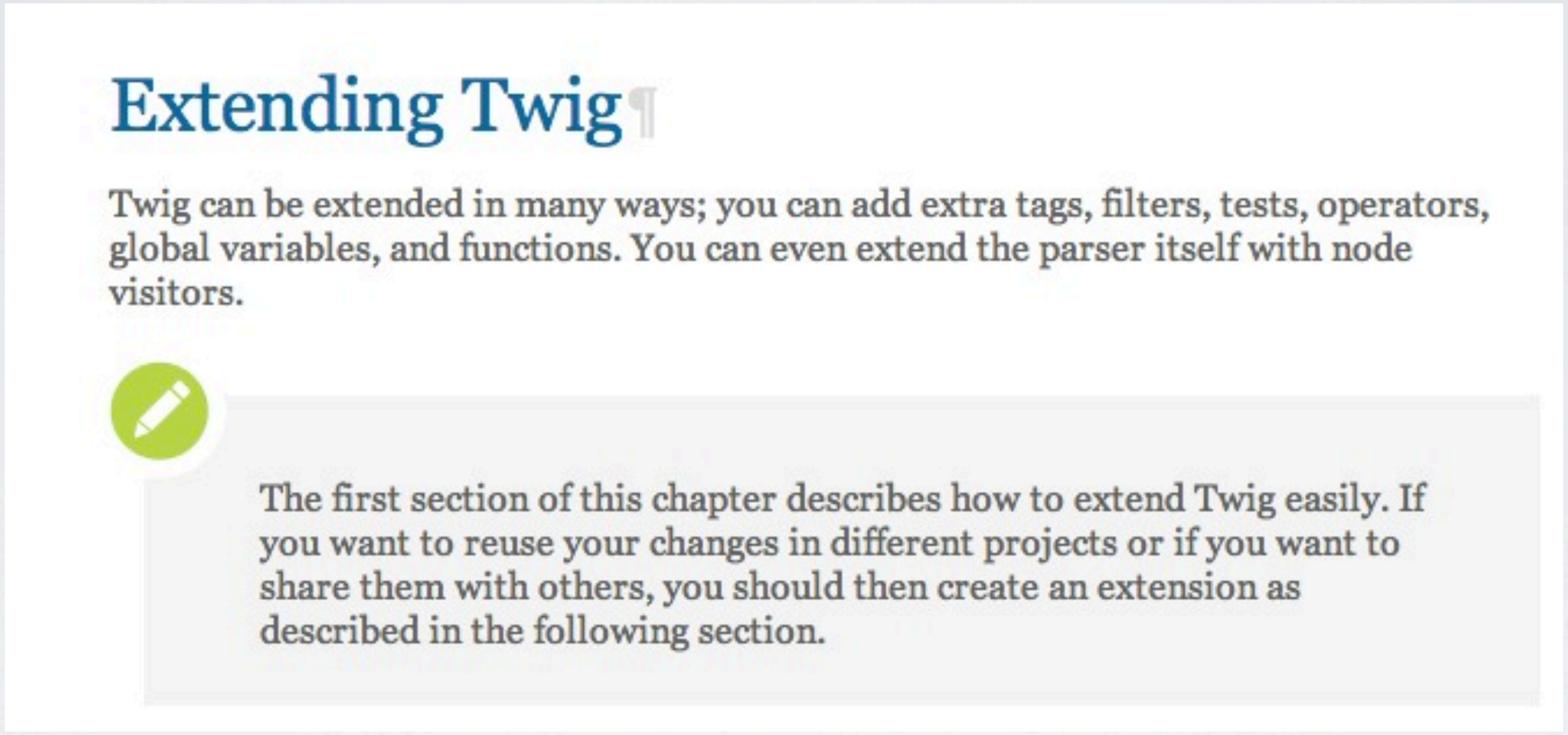
You can download the documentation for offline reading:

Twig Reference

Browse the online reference to learn more about built-in features.

well documented

TWIG



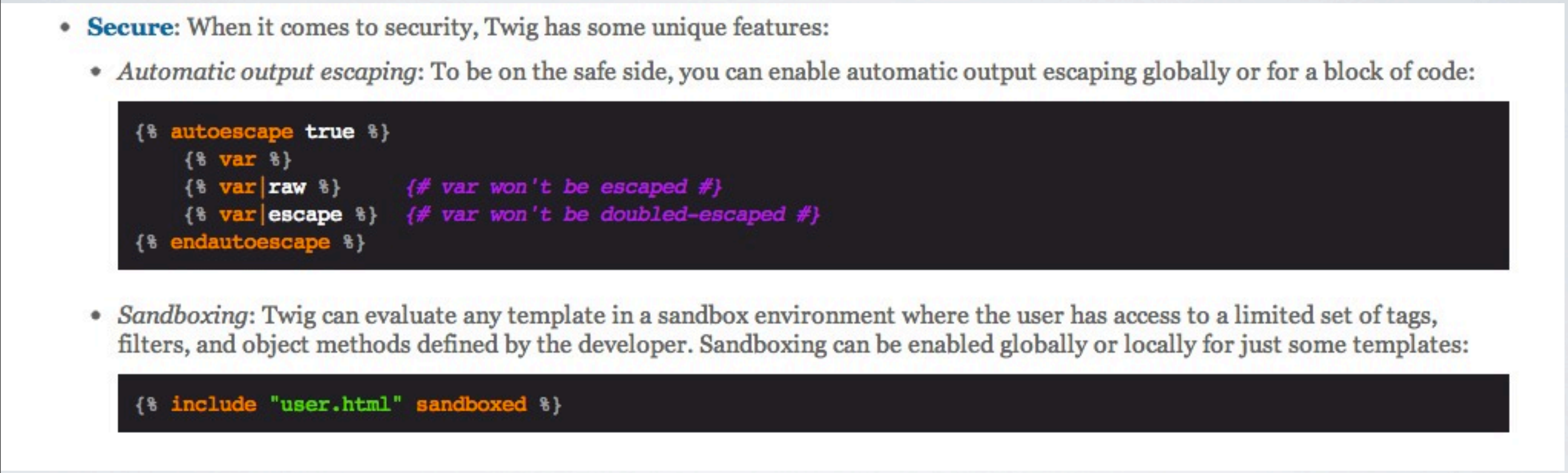
Extending Twig¶

Twig can be extended in many ways; you can add extra tags, filters, tests, operators, global variables, and functions. You can even extend the parser itself with node visitors.

The first section of this chapter describes how to extend Twig easily. If you want to reuse your changes in different projects or if you want to share them with others, you should then create an extension as described in the following section.

extensible

TWIG



- **Secure**: When it comes to security, Twig has some unique features:
  - *Automatic output escaping*: To be on the safe side, you can enable automatic output escaping globally or for a block of code:

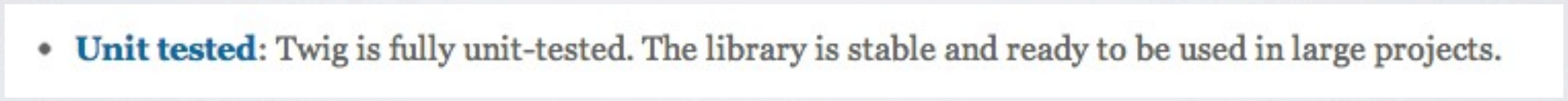
```
{% autoescape true %}
    {% var %}
    {% var|raw %}     {# var won't be escaped #}
    {% var|escape %}  {# var won't be doubled-escaped #}
{% endautoescape %}
```

  - *Sandboxing*: Twig can evaluate any template in a sandbox environment where the user has access to a limited set of tags, filters, and object methods defined by the developer. Sandboxing can be enabled globally or locally for just some templates:

```
{% include "user.html" sandboxed %}
```

secure

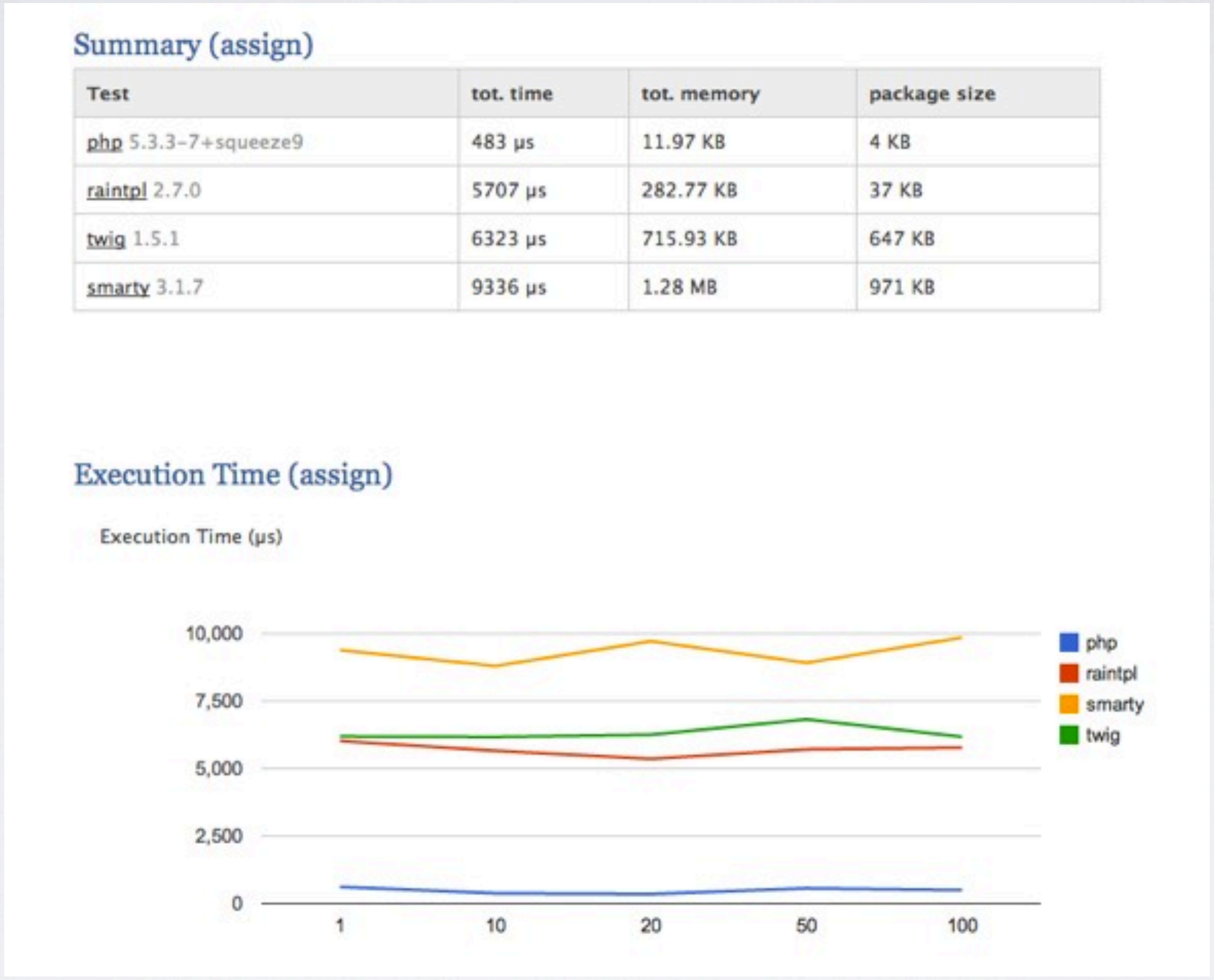
TWIG



- **Unit tested**: Twig is fully unit-tested. The library is stable and ready to be used in large projects.

well-tested

TWIG



Summary (assign)

Test	tot. time	tot. memory	package size
php 5.3.3-7+squeeze9	483 μs	11.97 KB	4 KB
raintpl 2.7.0	5707 μs	282.77 KB	37 KB
twig 1.5.1	6323 μs	715.93 KB	647 KB
smarty 3.1.7	9336 μs	1.28 MB	971 KB

Execution Time (assign)

fast

TWIG

IDE integration

TWIG

python

smarty
TEMPLATE ENGINE

django

recognizable syntax

TWIG

by Symfony's author, Fabien Potencier

TWIG

Principals to guide us.
BADCamp, 2012.

TWIG

Principals to guide us

1. Start with nothing

Core default markup should strive for semantic simplicity, with few HTML elements, added only as needed

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**

   Don't assume you know what people want or add features based on "What-if?" Think about the 90% of use cases.

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**
3. **Provide tools**

   Give front-end experts a way to achieve specific goals; goals that apply to the remaining 10% of use cases. Keep in mind that complex problems may require complex solutions.

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**
3. **Provide tools**
4. **Consolidate**
   "Your markup is not special." Don't make something new. Work toward common theme functions that modules leverage (i.e. a Theme Component Library).

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**
3. **Provide tools**
4. **Consolidate**
5. **Visibility**

   You should be able to see what's going on in a template without reading docs. Twig provides a lot of this by virtue of its syntax (it looks like HTML). Form follows function.

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**
3. **Provide tools**
4. **Consolidate**
5. **Visibility**
6. **Consistency**

   Do the same things everywhere, follow patterns. Use similar variable names across templates if they represent similar things.

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**
3. **Provide tools**
4. **Consolidate**
5. **Visibility**
6. **Consistency**
7. **Don't dumb it down**
   Complexity should be reduced but not obscured. Themers *can* understand template logic and loops. When complexity does happen, use comments to explain why.

TWIG

Principals to guide us

1. **Start with nothing**
2. **Build from use cases**
3. **Provide tools**
4. **Consolidate**
5. **Visibility**
6. **Consistency**
7. **Don't dumb it down**
8. **Organization should be driven by meaning and semantics over technical convenience**

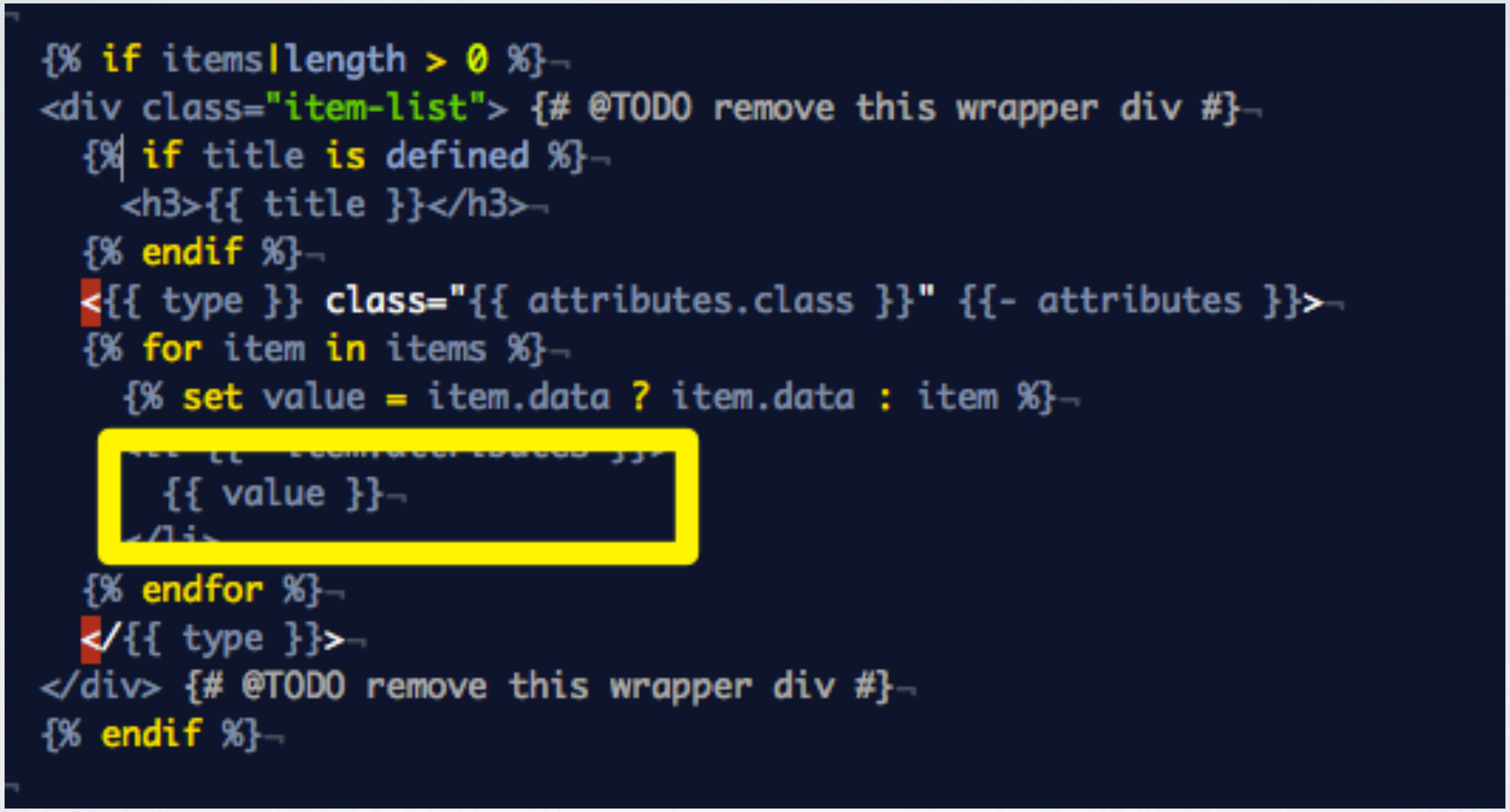
   Consider what an element means rather than how it structurally appears. Names and locations of templates should be self-evident. Themers want to see markup in templates, not abstraction.

TWIG

what does it look like?

TWIG

what does it look like?

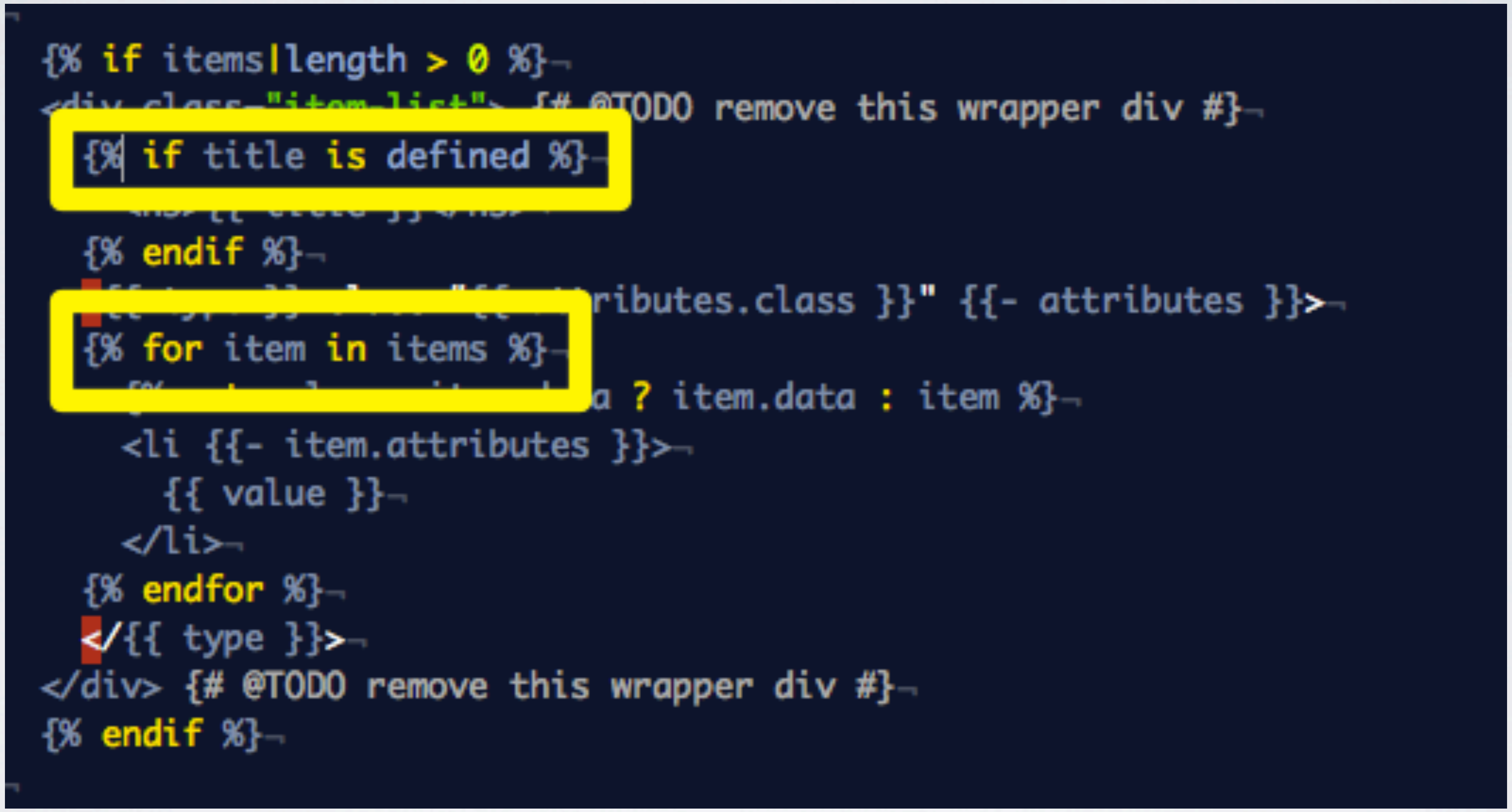


```
{% if items|length > 0 %}
<div class="item-list"> {# @TODO remove this wrapper div #}
  {% if title is defined %}
    <h3>{{ title }}</h3>
  {% endif %}
  <{{ type }} class="{{ attributes.class }}" {{- attributes }}>
  {% for item in items %}
    {% set value = item.data ? item.data : item %}
    <li {{ item.attributes }}>
      {{ value }}
    </li>
  {% endfor %}
  </{{ type }}>
</div> {# @TODO remove this wrapper div #}
{% endif %}
```

print with {{ }}

TWIG

what does it look like?

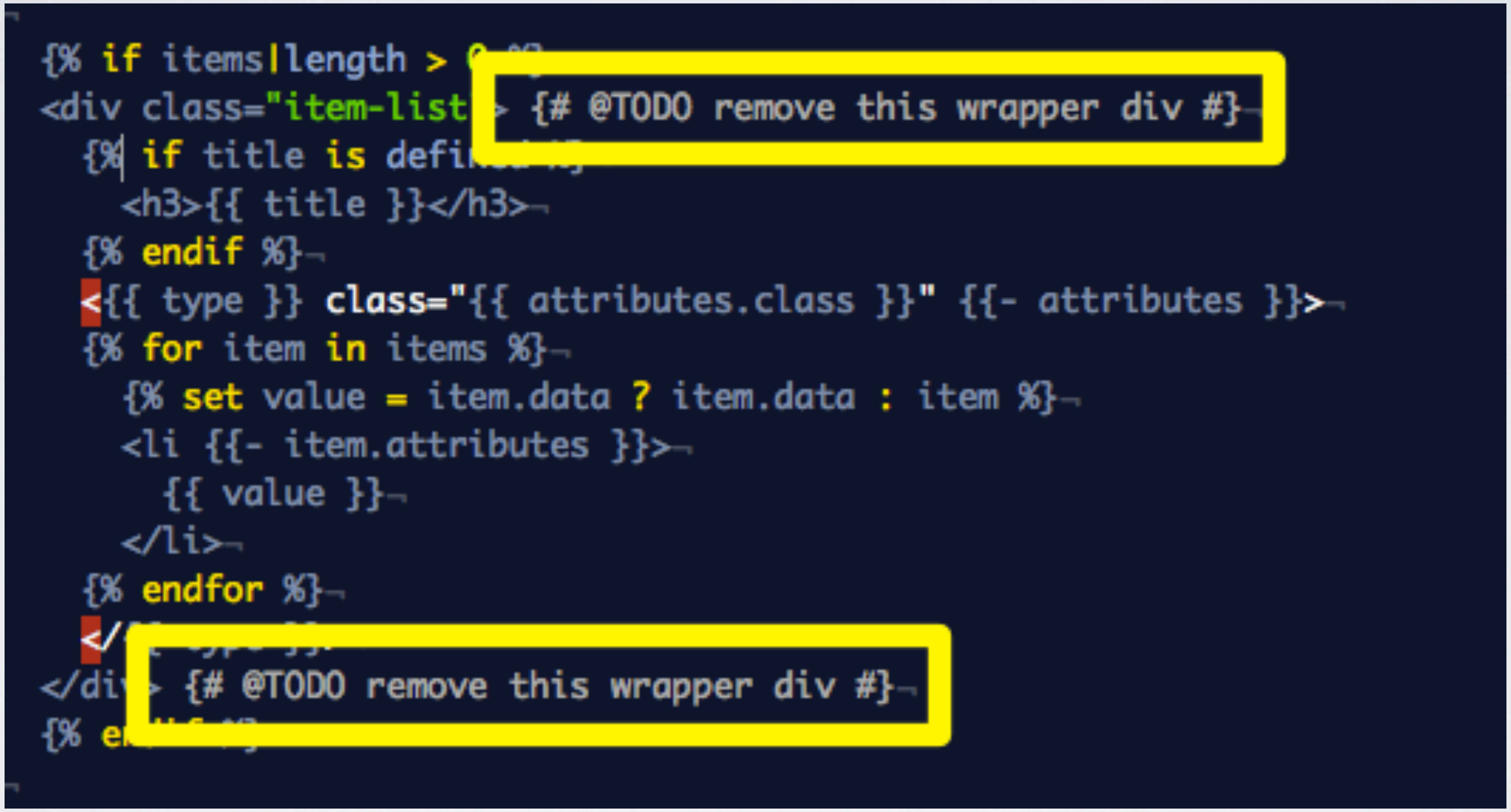


```
{% if items|length > 0 %}
<div class="item-list"> {# @TODO remove this wrapper div #}
  {% if title is defined %}
    <h3>{{ title }}</h3>
  {% endif %}
  <{{ type }} class="{{ attributes.class }}" {{- attributes }}>
  {% for item in items %}
    {% set value = item.data ? item.data : item %}
    <li {{- item.attributes }}>
      {{ value }}
    </li>
  {% endfor %}
  </{{ type }}>
</div> {# @TODO remove this wrapper div #}
{% endif %}
```

commands with {% %}

TWIG

what does it look like?

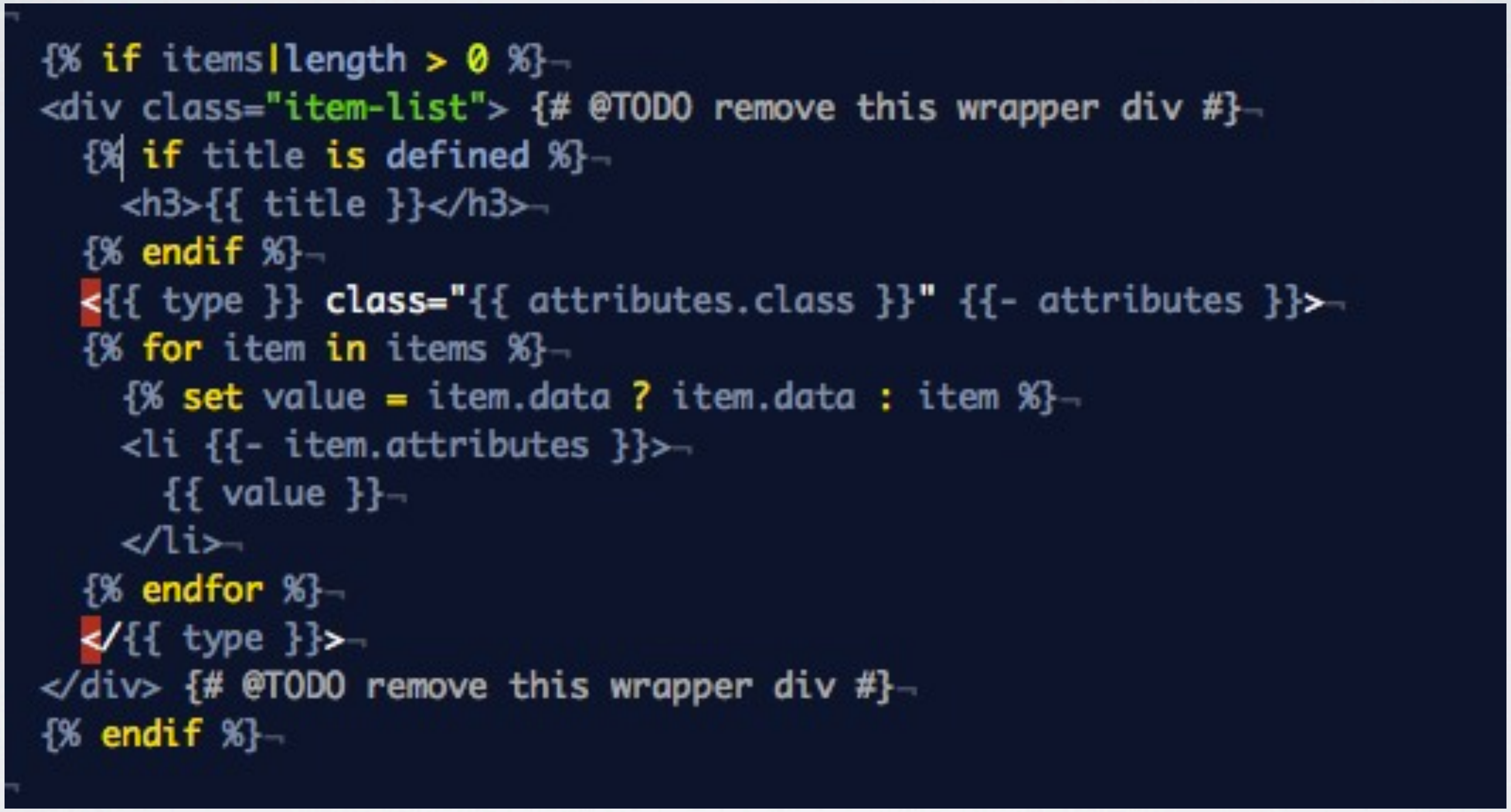


```
{% if items|length > 0 %}
<div class="item-list"> {# @TODO remove this wrapper div #}
  {% if title is defined %}
    <h3>{{ title }}</h3>
  {% endif %}
  <{{ type }} class="{{ attributes.class }}" {{- attributes }}>
  {% for item in items %}
    {% set value = item.data ? item.data : item %}
    <li {{- item.attributes }}>
      {{ value }}
    </li>
  {% endfor %}
  </{{ type }}>
</div> {# @TODO remove this wrapper div #}
{% endif %}
```

comments with {# #}

TWIG

what does it look like?



```
{% if items|length > 0 %}
<div class="item-list"> {# @TODO remove this wrapper div #}
  {% if title is defined %}
    <h3>{{ title }}</h3>
  {% endif %}
  <{{ type }} class="{{ attributes.class }}" {{- attributes }}>
  {% for item in items %}
    {% set value = item.data ? item.data : item %}
    <li {{- item.attributes }}>
      {{ value }}
    </li>
  {% endfor %}
  </{{ type }}>
</div> {# @TODO remove this wrapper div #}
{% endif %}
```

simple and intuitive

TWIG

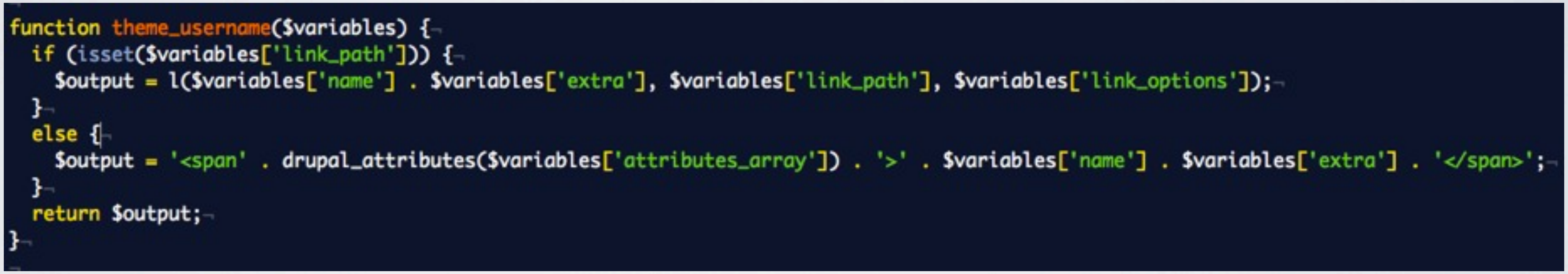
less code than PHP

all theme functions become template files.
a single way to override markup!

TWIG

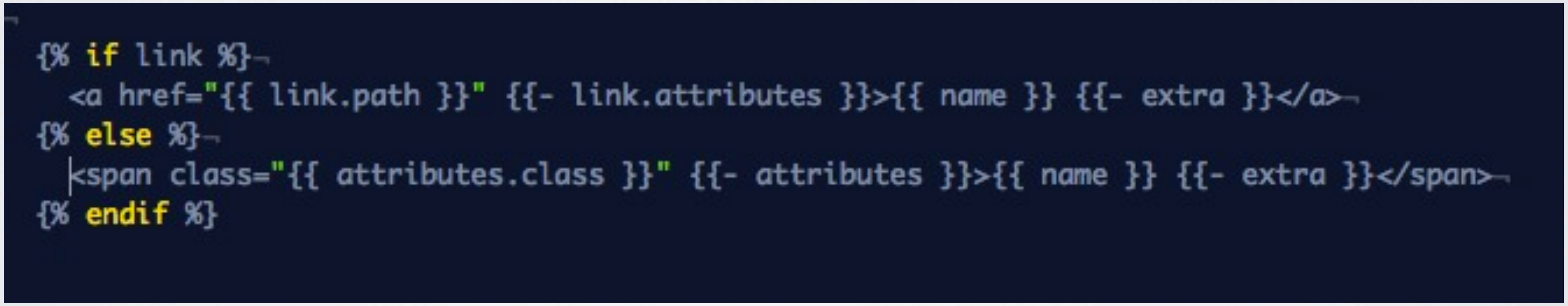
less code than PHP

D7



```
function theme_username($variables) {
  if (isset($variables['link_path'])) {
    $output = l($variables['name'] . $variables['extra'], $variables['link_path'], $variables['link_options']);
  }
  else {
    $output = '<span' . drupal_attributes($variables['attributes_array']) . '>' . $variables['name'] . $variables['extra'] . '</span>';
  }
  return $output;
}
```

D8



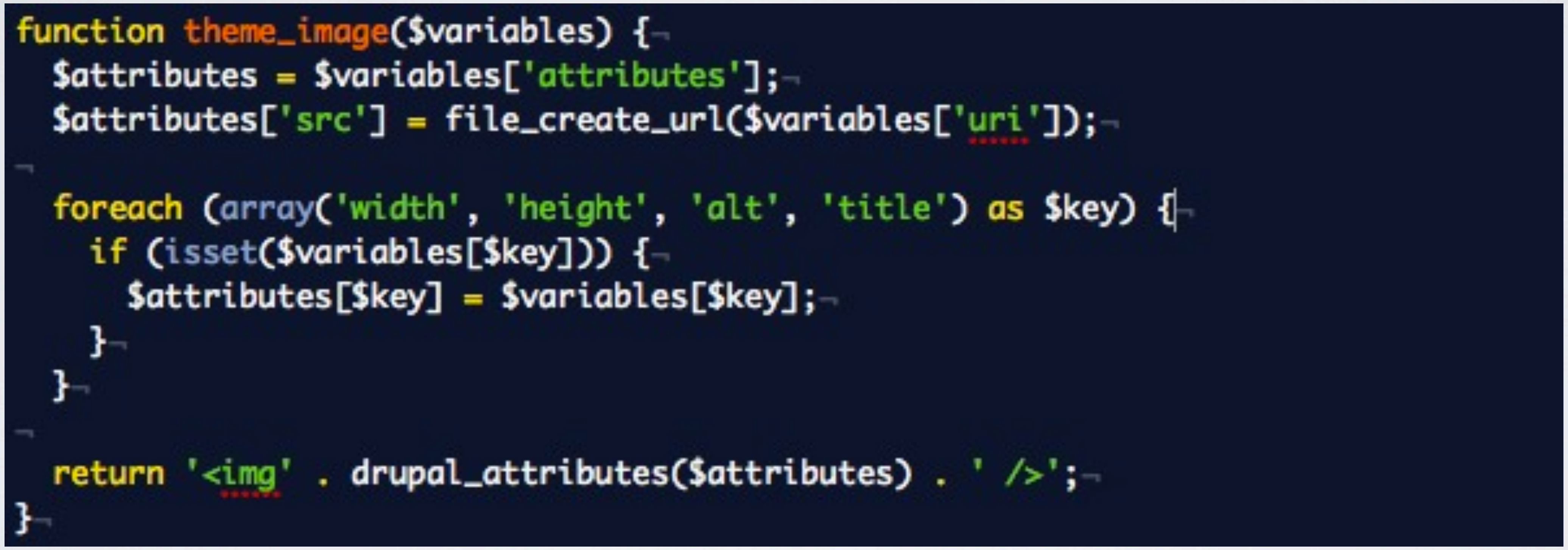
```
{% if link %}
  <a href="{{ link.path }}" {{- link.attributes }}>{{ name }} {{- extra }}</a>
{% else %}
  <span class="{{ attributes.class }}" {{- attributes }}>{{ name }} {{- extra }}</span>
{% endif %}
```

theme_username becomes username.html.twig

TWIG

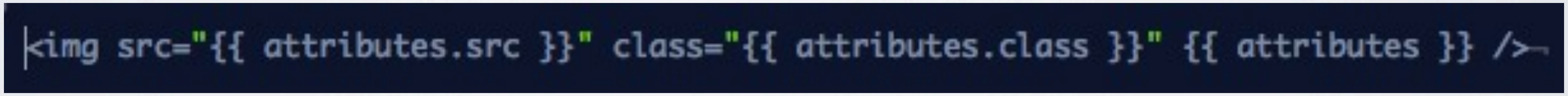
less code than PHP

D7



```
function theme_image($variables) {
  $attributes = $variables['attributes'];
  $attributes['src'] = file_create_url($variables['uri']);

  foreach (array('width', 'height', 'alt', 'title') as $key) {
    if (isset($variables[$key])) {
      $attributes[$key] = $variables[$key];
    }
  }

  return '<img' . drupal_attributes($attributes) . ' />';
}
```

D8



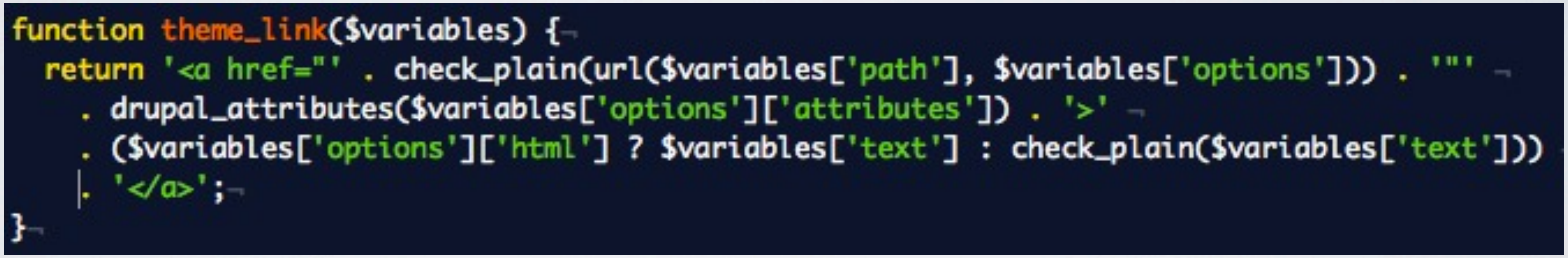
```
<img src="{{ attributes.src }}" class="{{ attributes.class }}" {{ attributes }} />
```

theme_image becomes image.html.twig

TWIG

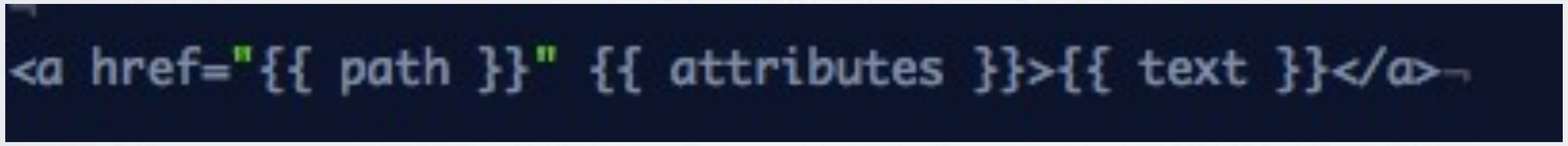
less code than PHP

D7



```
function theme_link($variables) {
  return '<a href="' . check_plain(url($variables['path'], $variables['options'])) . '"'
    . drupal_attributes($variables['options']['attributes']) . '>'
    . ($variables['options']['html'] ? $variables['text'] : check_plain($variables['text']))
    . '</a>';
}
```

D8



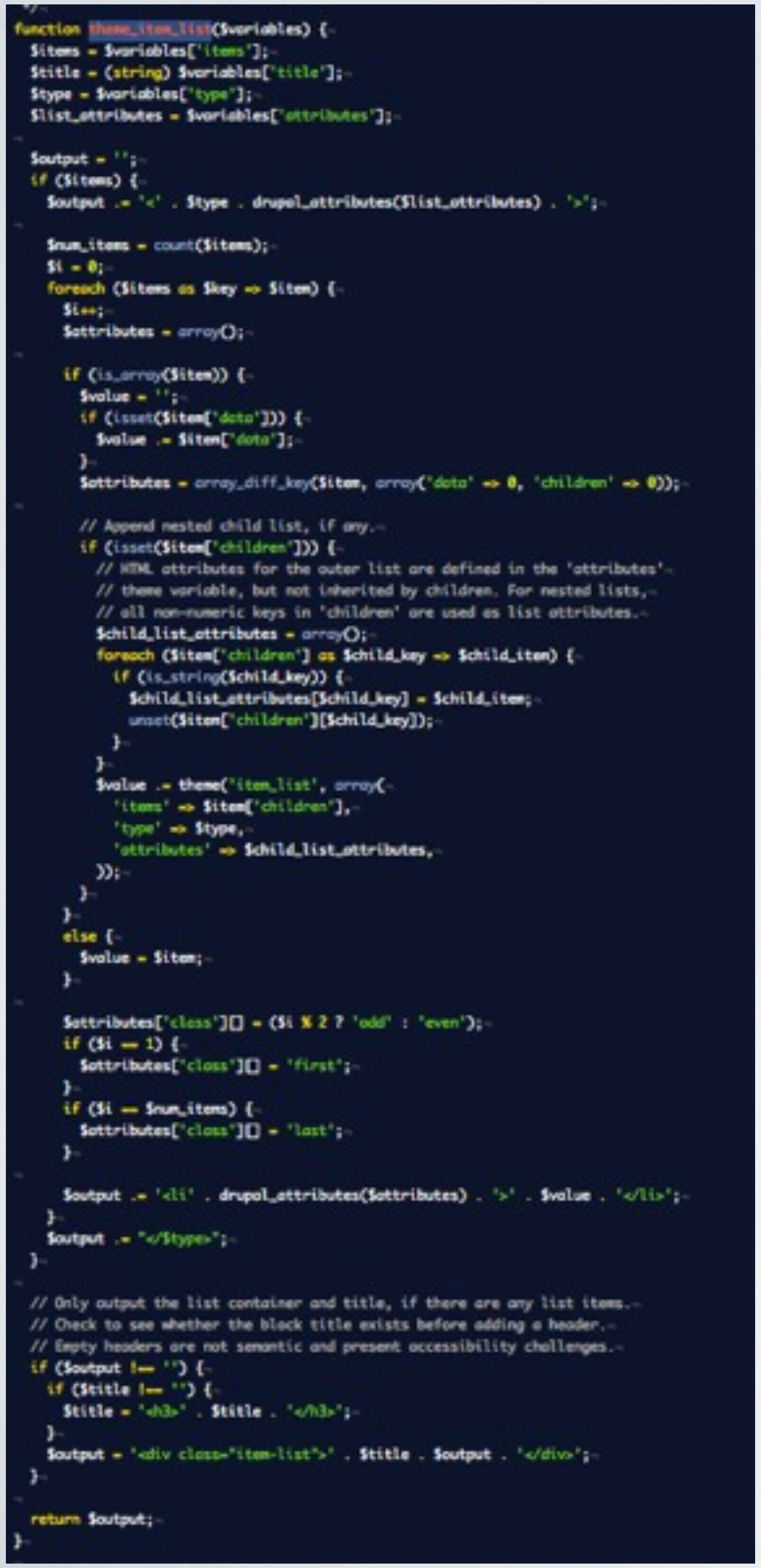
```
<a href="{{ path }}" {{ attributes }}>{{ text }}</a>
```

theme_link becomes link.html.twig

TWIG

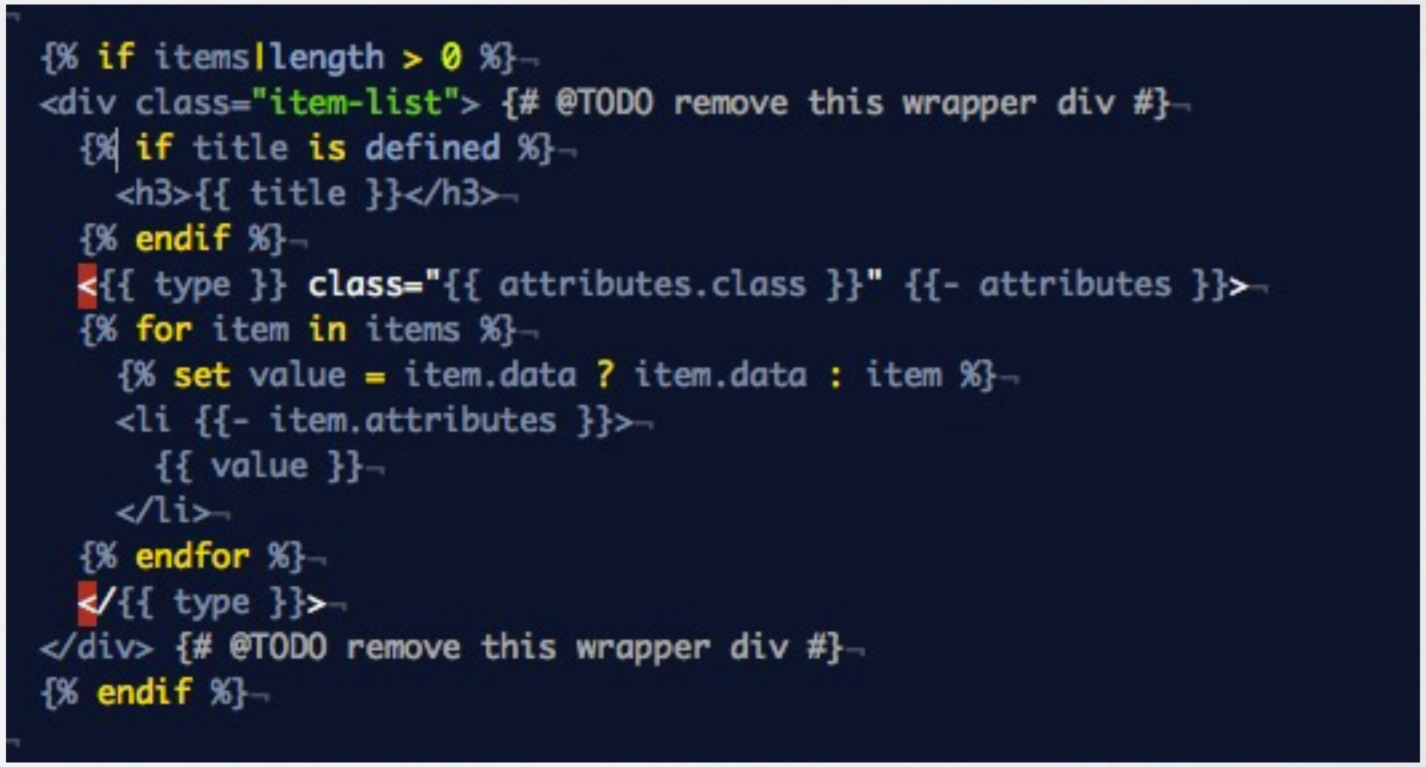
less code than PHP

D7



```php
 */
function theme_item_list($variables) {
  $items = $variables['items'];
  $title = (string) $variables['title'];
  $type = $variables['type'];
  $list_attributes = $variables['attributes'];

  $output = '';
  if ($items) {
    $output .= '<' . $type . drupal_attributes($list_attributes) . '>';

    $num_items = count($items);
    $i = 0;
    foreach ($items as $key => $item) {
      $i++;
      $attributes = array();

      if (is_array($item)) {
        $value = '';
        if (isset($item['data'])) {
          $value .= $item['data'];
        }
        $attributes = array_diff_key($item, array('data' => 0, 'children' => 0));

        // Append nested child list, if any.
        if (isset($item['children'])) {
          // HTML attributes for the outer list are defined in the 'attributes'
          // theme variable, but not inherited by children. For nested lists,
          // all non-numeric keys in 'children' are used as list attributes.
          $child_list_attributes = array();
          foreach ($item['children'] as $child_key => $child_item) {
            if (is_string($child_key)) {
              $child_list_attributes[$child_key] = $child_item;
              unset($item['children'][$child_key]);
            }
          }
          $value .= theme('item_list', array(
            'items' => $item['children'],
            'type' => $type,
            'attributes' => $child_list_attributes,
          ));
        }
      }
      else {
        $value = $item;
      }

      $attributes['class'][] = ($i % 2 ? 'odd' : 'even');
      if ($i == 1) {
        $attributes['class'][] = 'first';
      }
      if ($i == $num_items) {
        $attributes['class'][] = 'last';
      }

      $output .= '<li' . drupal_attributes($attributes) . '>' . $value . '</li>';
    }
    $output .= "</$type>";
  }

  // Only output the list container and title, if there are any list items.
  // Check to see whether the block title exists before adding a header.
  // Empty headers are not semantic and present accessibility challenges.
  if ($output !== '') {
    if ($title !== '') {
      $title = '<h3>' . $title . '</h3>';
    }
    $output = '<div class="item-list">' . $title . $output . '</div>';
  }

  return $output;
}
```

D8



```twig
{% if items|length > 0 %}
<div class="item-list"> {# @TODO remove this wrapper div #}
  {% if title is defined %}
    <h3>{{ title }}</h3>
  {% endif %}
  <{{ type }} class="{{ attributes.class }}" {{- attributes }}>
  {% for item in items %}
    {% set value = item.data ? item.data : item %}
    <li {{- item.attributes }}>
      {{ value }}
    </li>
  {% endfor %}
  </{{ type }}>
</div> {# @TODO remove this wrapper div #}
{% endif %}
```

theme_item_list beomes item_list.html.twig

REMEMBER DRUPAL 7?

- Drupal-specific template conventions
- Mixed data types (strings, objects & arrays)

  $node->nid vs $content['links']
- Different methods of printing

  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure
- Two ways to override markup

  Templates: *.tpl.php vs functions: theme_foo()
- Many template files, many similar theme functions
- Complex mix of subsystems
- Difficult to learn for newcomers (and D6 veterans)

REMEMBER DRUPAL 7?

- Drupal-specific template conventions

Twig is used elsewhere on the web.
...and looks a lot more like HTML.

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays)

    $node->nid vs $content[‘links’]

All template variables are accessed consistently:

node.nid

content.links

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**

  $node->nid vs $content['links']
- Different methods of printing

  print $node->nid vs print render($content['links'])

We're removed calls to render() from templates:

{{ node.nid }}

{{ content.links }}

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**

  $node->nid vs $content['links']
- Different methods of printing **FIXED**

  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure

All variables will be *automatically* sanitized.

...and most PHP functions cannot be executed in template files.

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**

  $node->nid vs $content[‘links’]
- Different methods of printing **FIXED**

  print $node->nid vs print render($content[‘links’])
- PHPTemplate is insecure **FIXED**
- Two ways to override markup

  Templates: *.tpl.php vs functions: theme_foo()

All theme functions are converted to template files

node.tpl.php becomes node.html.twig

theme_table() becomes table.html.twig

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**

  $node->nid vs $content['links']
- Different methods of printing **FIXED**

  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure **FIXED**
- Two ways to override markup **FIXED**

  Templates: *.tpl.php vs functions: theme_foo()
- Many template files, many similar theme functions

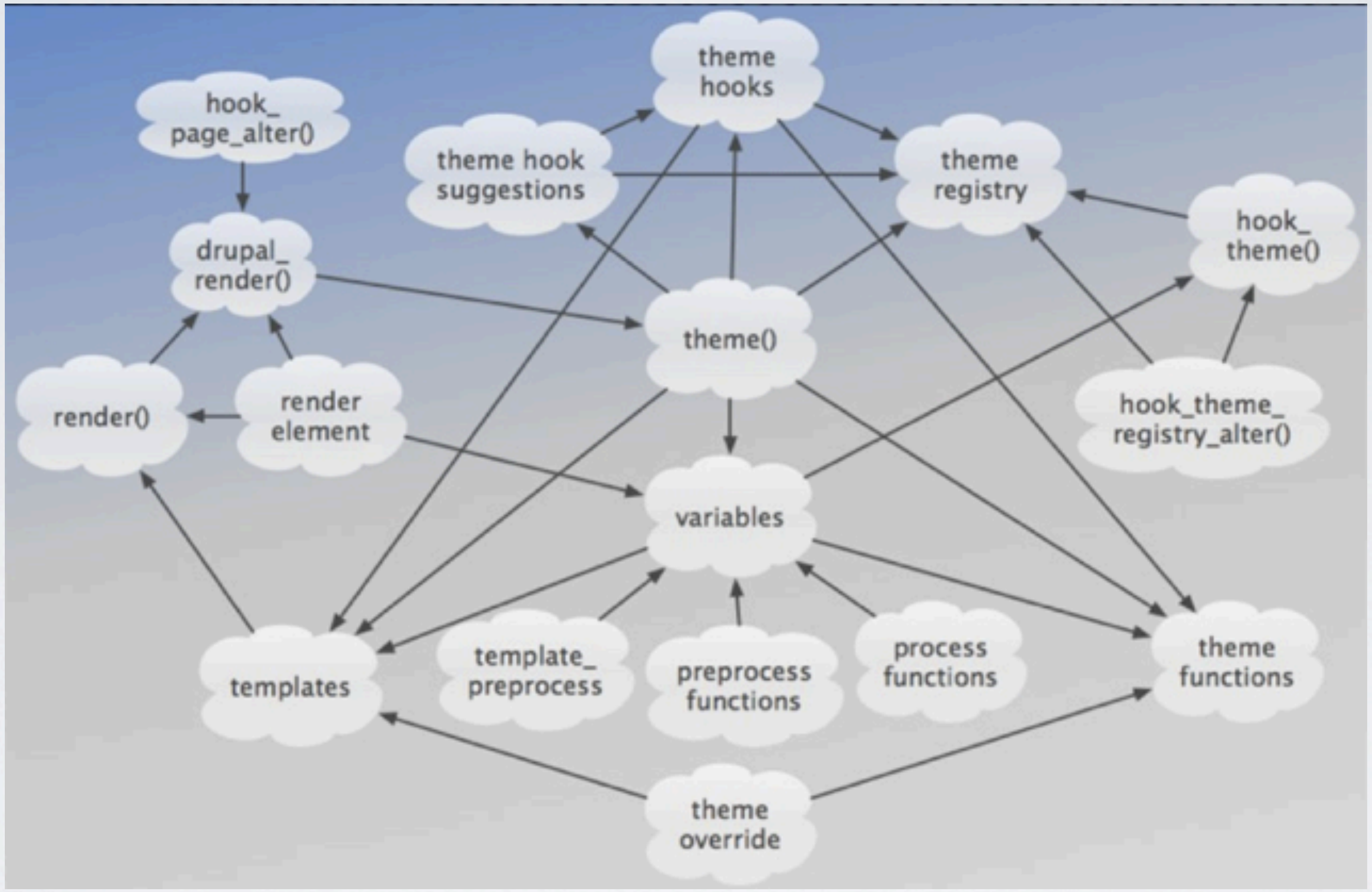
We're working on this right now (Sprints!)

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**
  $node->nid vs $content['links']
- Different methods of printing **FIXED**
  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure **FIXED**
- Two ways to override markup **FIXED**
  Templates: *.tpl.php vs functions: theme_foo()
- Many template files, many similar theme functions **@todo**
- Complex mix of subsystems

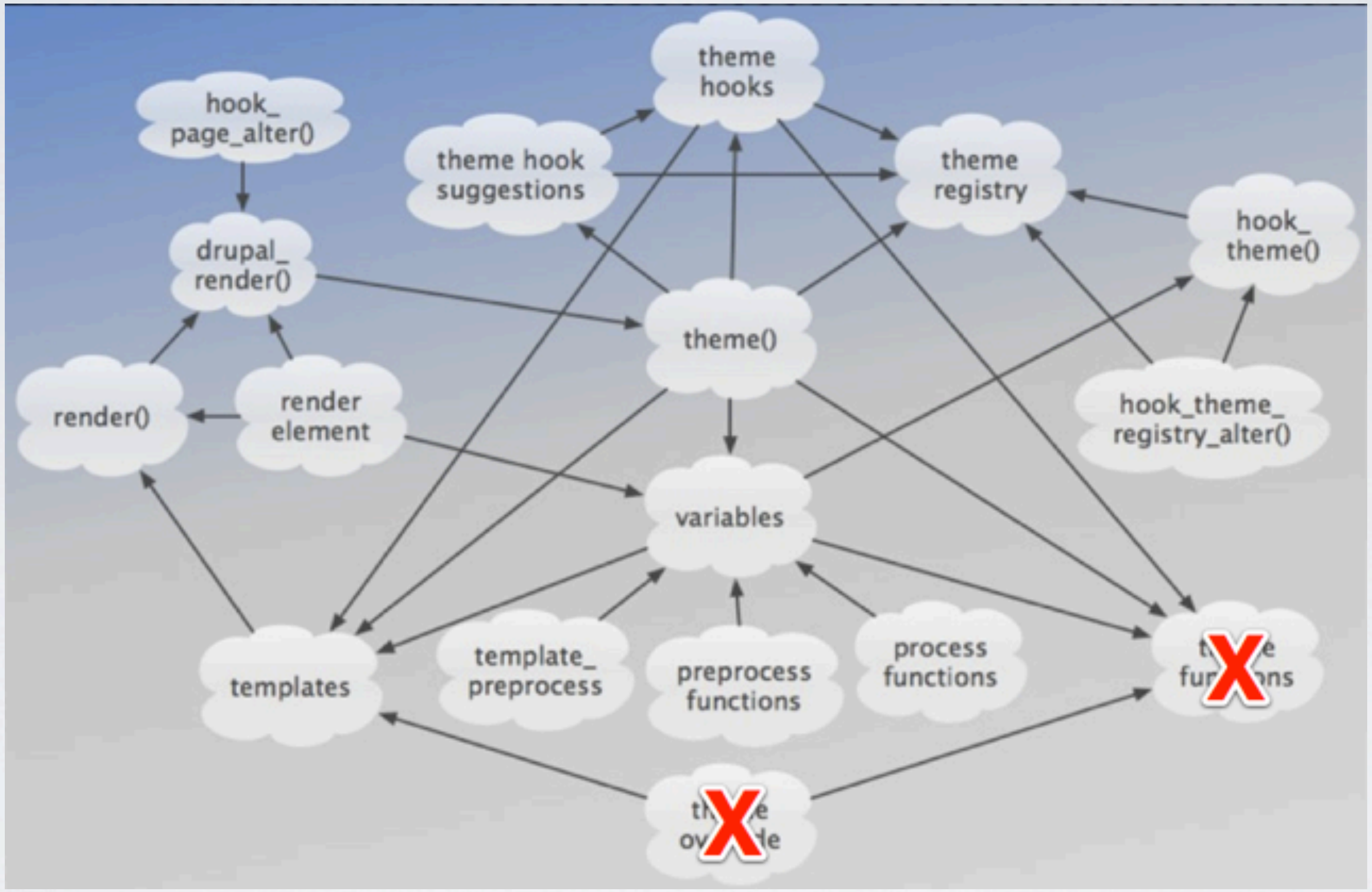
We can remove all theme functions, and potentially render, process & (maybe) preprocess.

REMEMBER DRUPAL 7?



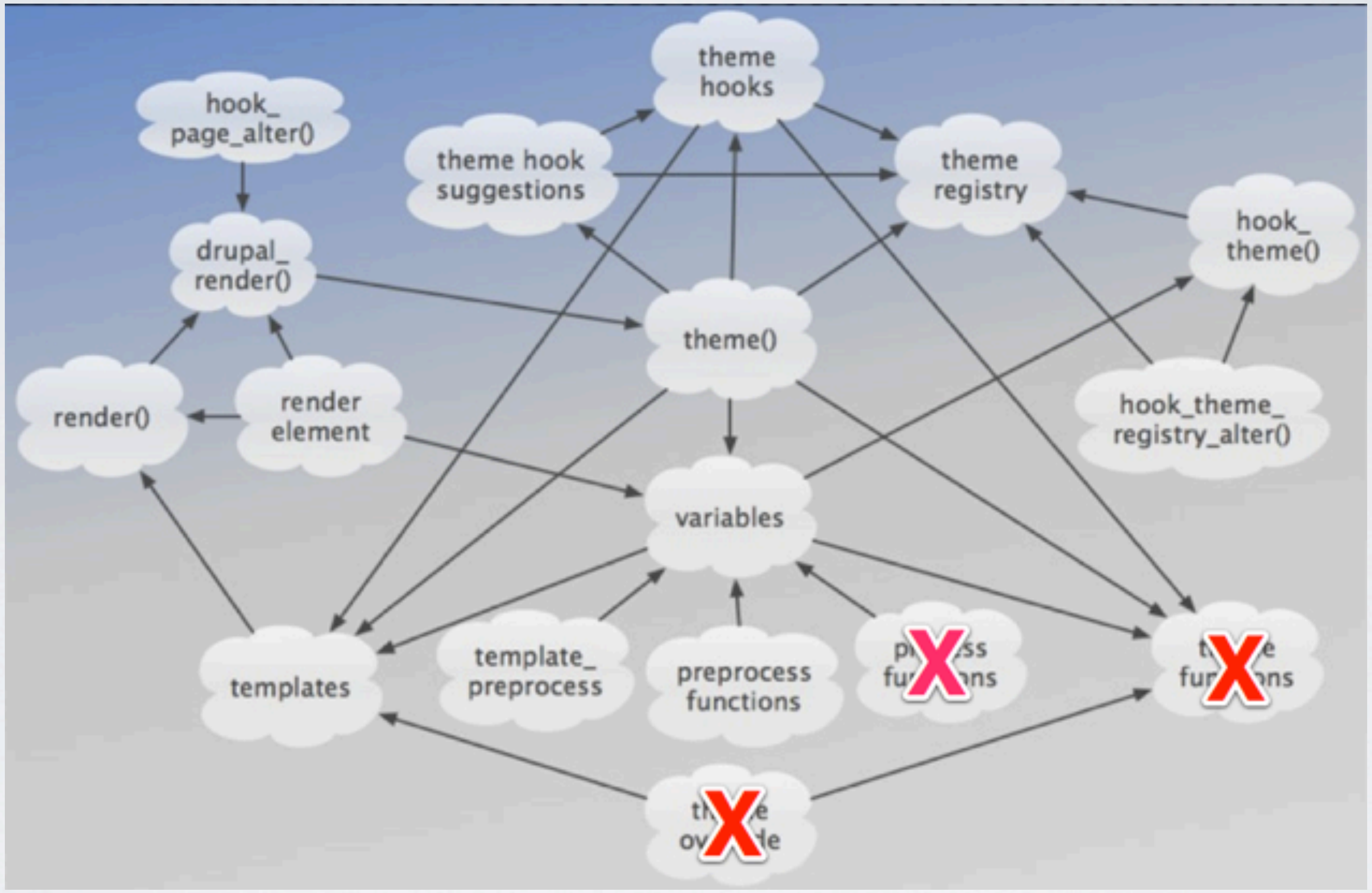
remember the complexity of Drupal 7?

REMEMBER DRUPAL 7?



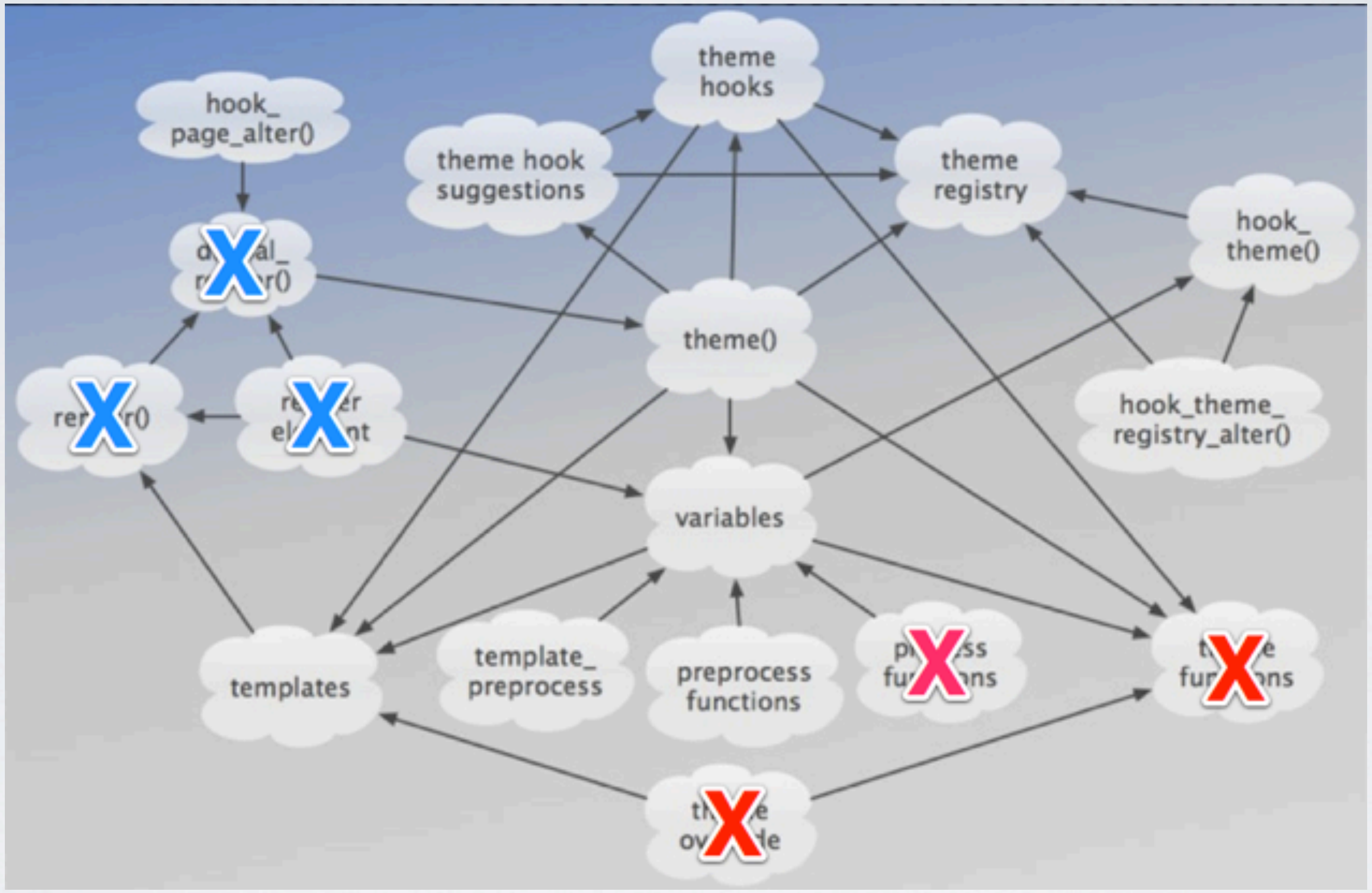
remove theme functions (and overrides) entirely.

REMEMBER DRUPAL 7?



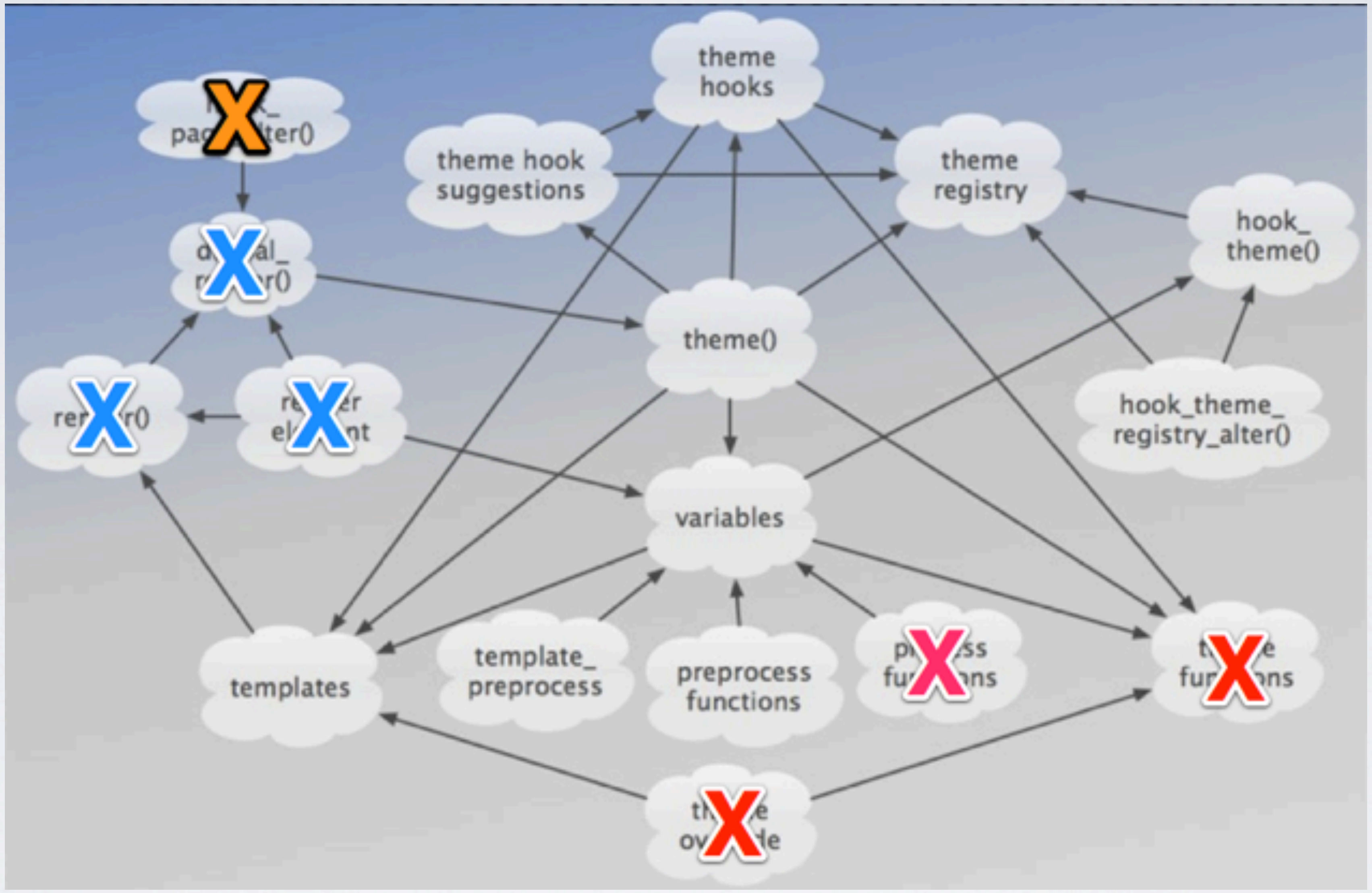
remove process.

REMEMBER DRUPAL 7?



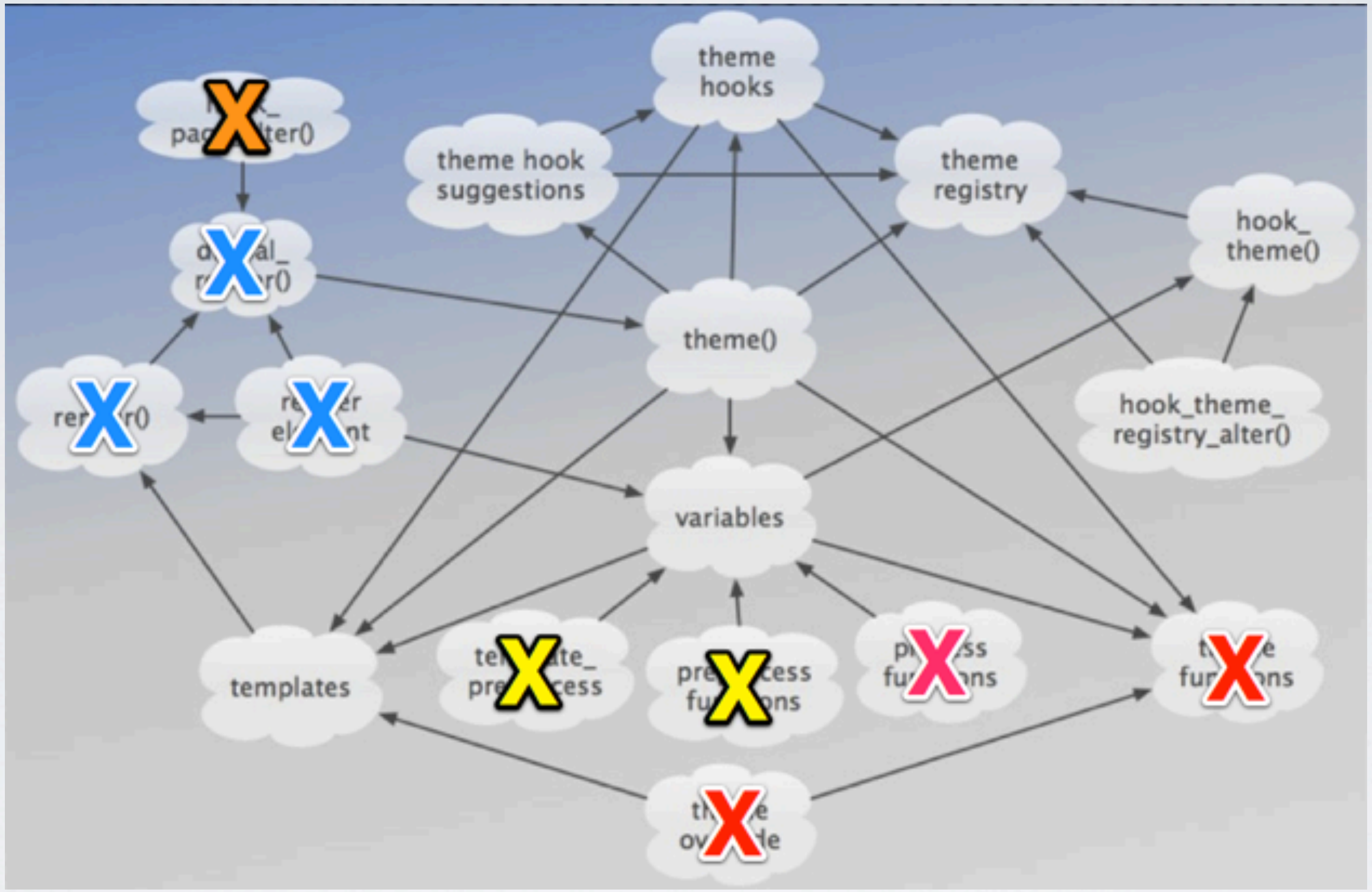
remove render.

REMEMBER DRUPAL 7?



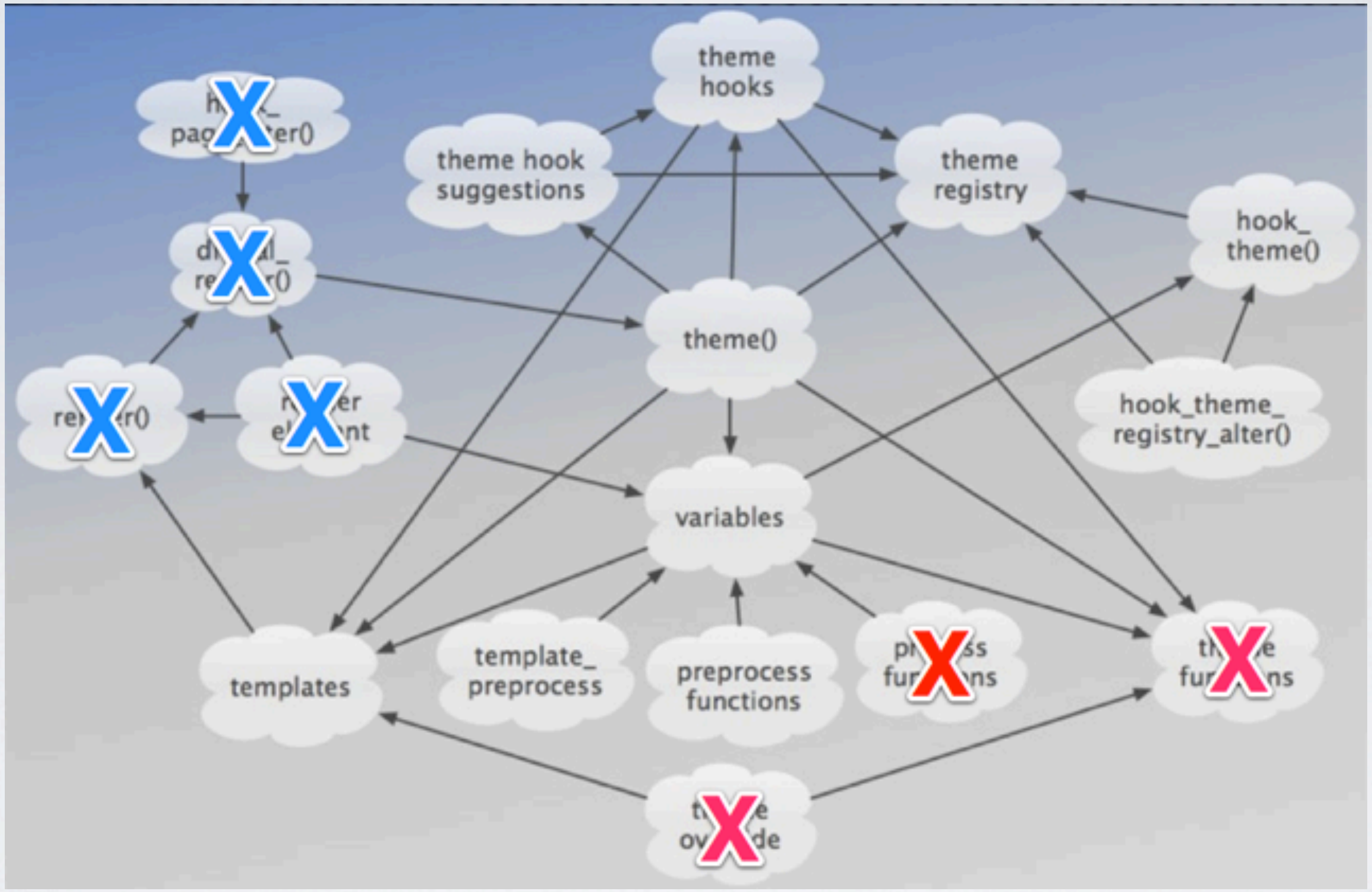
remove page alter?

REMEMBER DRUPAL 7?



remove preprocess?

REMEMBER DRUPAL 7?



look what would happen in Drupal 8.

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**

  $node->nid vs $content['links']
- Different methods of printing **FIXED**

  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure **FIXED**
- Two ways to override markup **FIXED**

  Templates: *.tpl.php vs functions: theme_foo()
- Many template files, many similar theme functions **@todo**
- Complex mix of subsystems **@todo**
- Difficult to learn for newcomers (and D6 veterans)

REMEMBER DRUPAL 7?

- Drupal-specific template conventions **FIXED**
- Mixed data types (strings, objects & arrays) **FIXED**

  $node->nid vs $content['links']
- Different methods of printing **FIXED**

  print $node->nid vs print render($content['links'])
- PHPTemplate is insecure **FIXED**
- Two ways to override markup **FIXED**

  Templates: *.tpl.php vs functions: theme_foo()
- Many template files, many similar theme functions **@todo**
- Complex mix of subsystems **@todo**
- ~~Difficult to learn for newcomers (and D6 veterans)~~

Consistency FTW

TWIG: OTHER WINS

- Twig template inheritance
- Variable inspection: Twig's dpm()
- Possible performance gains: (much TBD) faster than render()?
- 2-way communication between UI and code: The UI can understand what variables exist (or not) in a template (but, contrib only folks!)

HOW CAN I HELP?

NEXT TASKS

- ~~Finish converting all existing core *.tpl.php files to Twig templates (sandbox)~~ **DONE**
- Finish converting all existing core theme functions to Twig templates (sandbox) **CLOSE?**
- Create & test patches for each module (core) **HELP!**
- Help clean up markup in core (sandbox/core) **HELP!**
- Consolidate similar templates (core) **HELP!**
- Identify & enable template suggestions (core) **HELP!**
- Rethink preprocess / __to_string() methods (maybe?)
- Convert renderable arrays into Twig objects (maybe?)

- More TBD

SPRINT!

Twig Sprint: **Saturday** Feb 9th, Centenial Room
(http://sydney2013.drupal.org/program/code-sprint)

- Create & test patches for each module in core.
- Convert theme functions to template files.
- Help clean up markup in core.
- More!

Come talk to us!

QUESTIONS?

A NEW THEME LAYER FOR DRUPAL 8

photo credits:

lolcat-flexible
http://cheezburger.com/2679924736

anything is possible pebbles
http://www.invergordontours.com/aip.html

lolcat questionmark
http://icanhascheezburger.com/2007/10/31/11197/

wheel-reinvented
http://www.brainwads.net/drewhawkins/2012/01/dont-re-invent-the-wheel-make-something-better/

objects
http://2teachers1classroom.blogspot.com/2009_02_01_archive.html

shapes
http://englishclass.jp/reading/topic/For_Screening_Purposes_Only

secure
http://blog.stratepedia.org/2010/06/03/what-is-a-secure-site/

consistency
http://icsigns.org/press/2010/03/23/consistency-staying-on-the-mark/

twig bird comic
http://s302.photobucket.com/albums/nn105/walkseva/?action=view&current=thebirdneedsthattwig.gif&currenttag=bird%20park%20twig%20comic%20need%20it

twig docs screenshots
http://twig.sensiolabs.org/documentation

twig speed graphs
http://phpcomparison.net/

python icon
http://python-hosting.org/

ruby icon
http://itmediaconnect.ro/en/web

django logo
http://py-arahat.blogspot.com/2010/08/django-vs-pylons.html

symfony logo
http://symfony.com/logo

scotch glass
http://www.thespir.it/articles/scotch-101/?viewall=1