Writing Secure Code

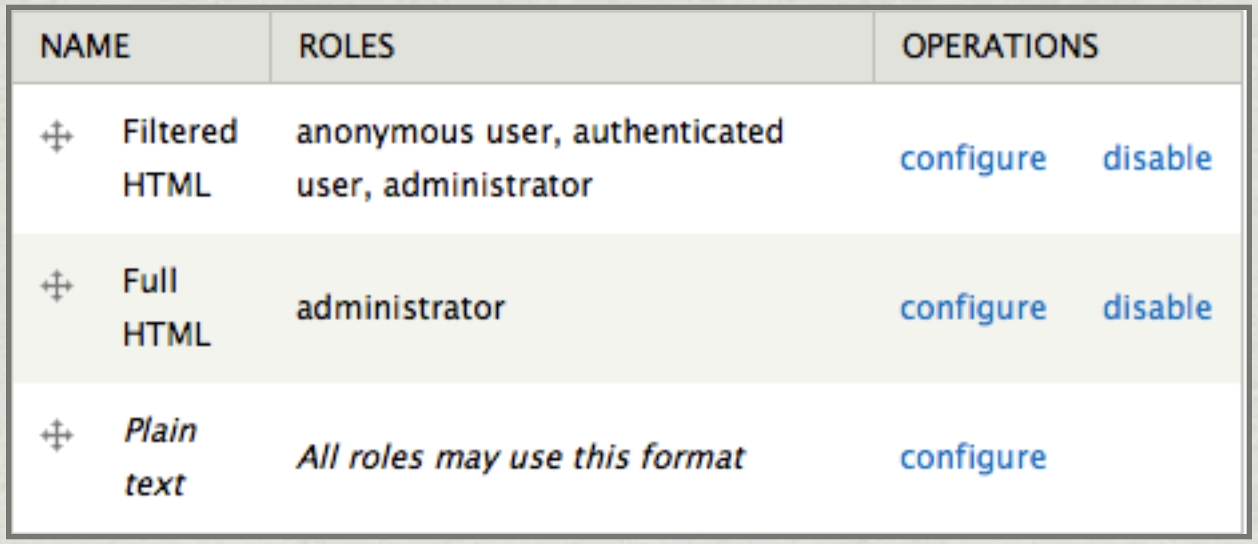
Wash all user-submitted data

Vulnerabilities

- XSS - Cross-site scripting
  Executing unauthorized code
- XSRF - Cross-site request forgery
  Remote execution for logged-in users
- SQL Injection
  A malicious variable placed into a query

How Drupal Handles Input

- Data is saved as-is into the database.
- Data gets 'washed' on output.

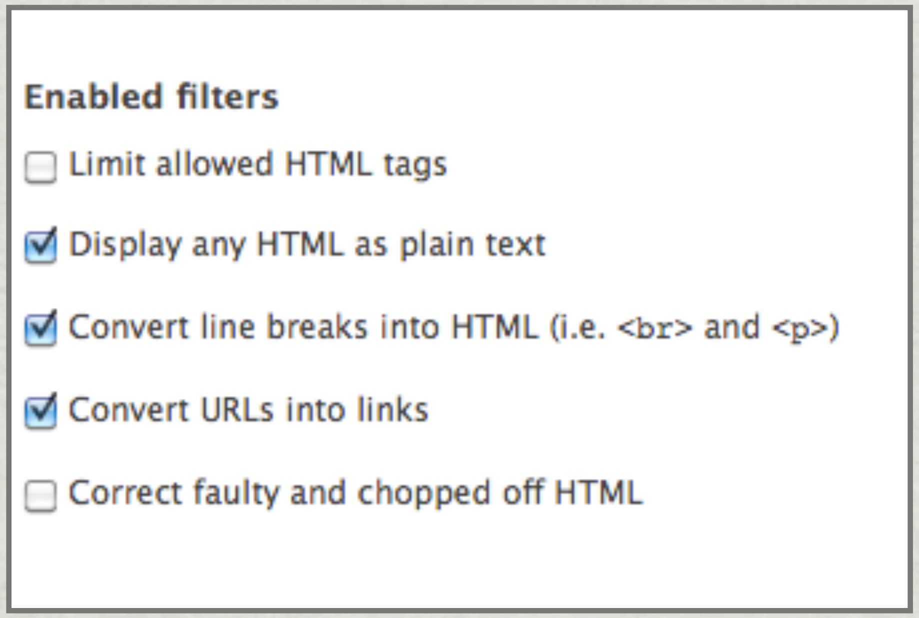


NAME	ROLES	OPERATIONS	
Filtered HTML	anonymous user, authenticated user, administrator	configure	disable
Full HTML	administrator	configure	disable
Plain text	*All roles may use this format*	configure	

- Burden lies with module and theme developers.
- Drupal gives you the tools. You must use them.

Text Formats

- /admin/config/content/formats
- Plain Text Filters
  - Display any HTML as plain text
    - <div class=”blue”>Hello!</div> becomes &lt;div class="blue"&gt;Hello!&lt;/div&gt;



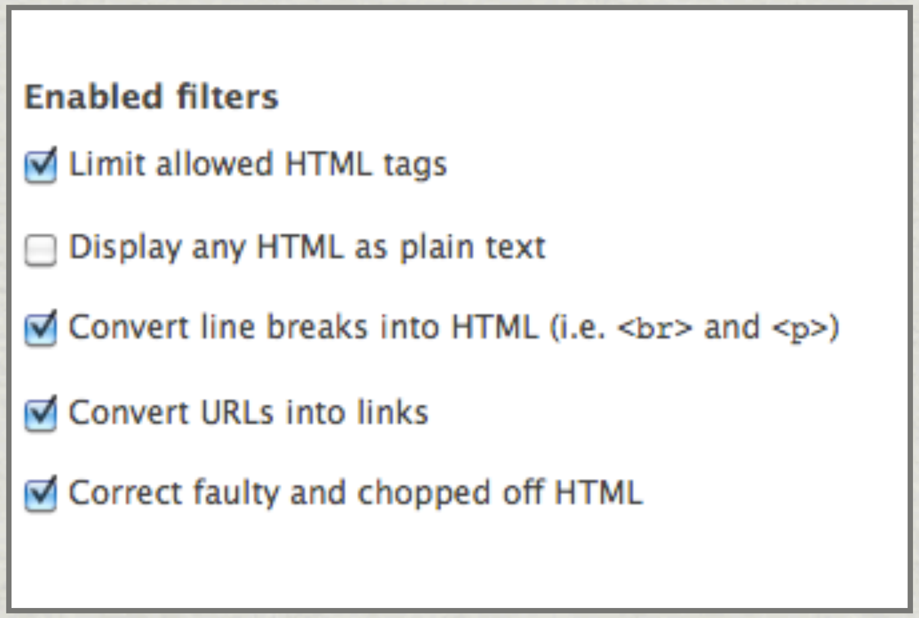
Enabled filters

- [ ] Limit allowed HTML tags
- [✓] Display any HTML as plain text
- [✓] Convert line breaks into HTML (i.e.
 and <p>)
- [✓] Convert URLs into links
- [ ] Correct faulty and chopped off HTML

drupalize.me

Text Formats

- /admin/config/content/formats
- Filtered HTML Filters
  - Limit allowed HTML tags
    - <a> <em> <strong> <cite> <blockquote> <code> <ul> <ol> <li> <dl> <dt> <dd>
    - JavaScript event attributes, JavaScript URLs, and CSS are always stripped.
    - <div class=”blue”>Hello!</div> becomes Hello!
      <code class="blue" style="border:1px;">Hello!</code> becomes
      <code class=”blue”>Hello!</code>



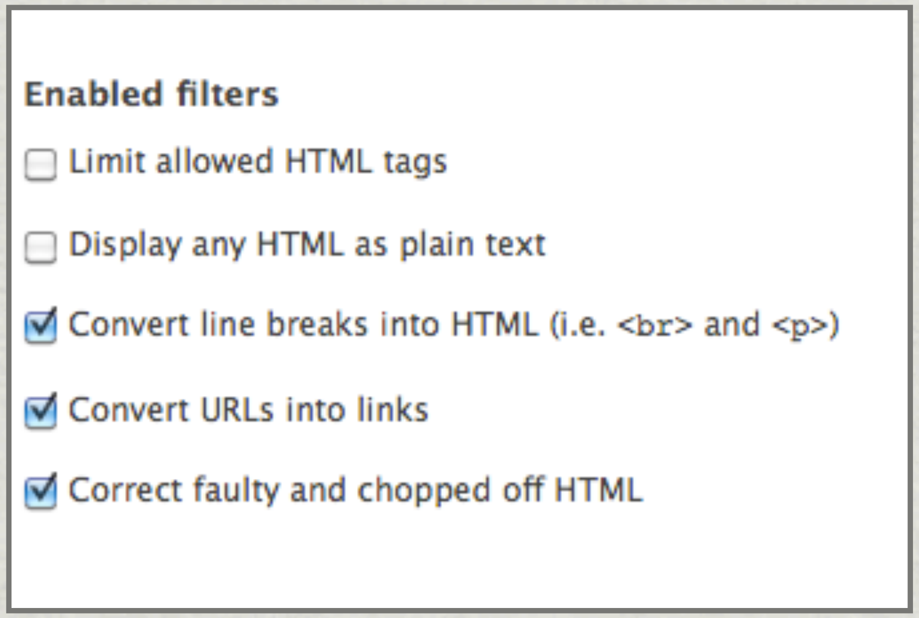
Enabled filters

- [✓] Limit allowed HTML tags
- [ ] Display any HTML as plain text
- [✓] Convert line breaks into HTML (i.e.
 and <p>)
- [✓] Convert URLs into links
- [✓] Correct faulty and chopped off HTML

drupalize.me

Text Formats

- /admin/config/content/formats
- Full HTML Filters

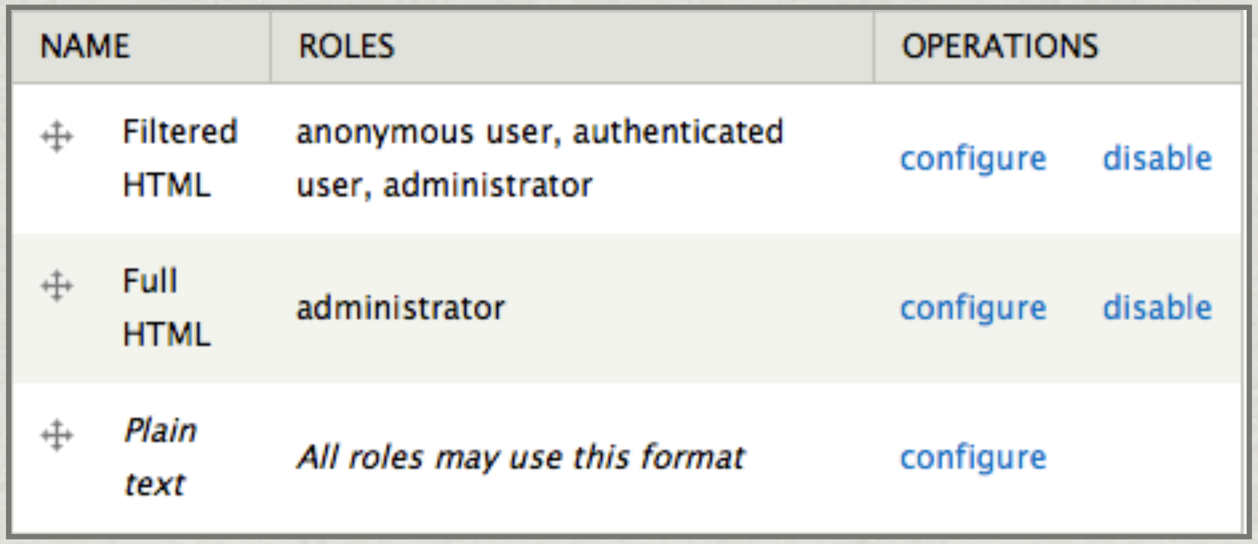


Enabled filters

- [ ] Limit allowed HTML tags
- [ ] Display any HTML as plain text
- [x] Convert line breaks into HTML (i.e. `<br>` and `<p>`)
- [x] Convert URLs into links
- [x] Correct faulty and chopped off HTML

How You Handle Output

- Burden lies with module and theme developers.
- Drupal gives you the tools. You must use them.



NAME	ROLES	OPERATIONS	
Filtered HTML	anonymous user, authenticated user, administrator	configure	disable
Full HTML	administrator	configure	disable
Plain text	*All roles may use this format*	configure	

Wash Your Output

- http://api.drupal.org/api/drupal/includes--common.inc/group/sanitization/7
- check_plain()
- check_markup()
- check_url() & l()
- t()
- filter_xss()
- filter_xss_admin()

check_plain($text)

- To be used when inserting plain text into HTML.
- No HTML is output.
- Uses the encoded special characters instead.
- <a href='test'>Test</a>
- &lt;a href=&#039;test&#039;&gt;Test&lt;/a&gt;

drupalize.me

check_markup($text, $format_id = NULL, $langcode = '', $cache = FALSE)

- To be used when inserting rich text into HTML.
- Allows you to specify the format ID that corresponds with the 'text format' you want to use.
- Falls back to the system default.
- $newtext = check_markup($text, 'filtered_html');

check_url($uri) l($text, $path, array $options = array())

- check_url() Strips dangerous protocols (e.g. 'javascript:') from a URI and encodes it for output to an HTML attribute value.
  - Allows 'ftp', 'http', 'https', 'irc', 'mailto', 'news', 'nntp', 'rtsp', 'sftp', 'ssh', 'tel', 'telnet', 'webcal'
- l() constructs a full HTML link utilizing url() for the href attribute and also sanitizes the link title.

t($string, $args = array(), $options = array())

- Allows strings to be translatable.
- Must use variable substitution for ANY variable.
- $text = t("@name's blog", array('@name' => format_username($account)));
  - !variable: Inserted as is. Use this for text that has already been sanitized.
  - @variable: Escaped to HTML using check_plain(). Use this for anything displayed on a page on the site.
  - %variable: Escaped as a placeholder for user-submitted content using drupal_placeholder(), which shows up as <em>emphasized</em> text.

filter_xss($string, $allowed_tags = array('a', 'em', 'strong', 'cite', 'blockquote', 'code', 'ul', 'ol', 'li', 'dl', 'dt', 'dd'))

- Filters an HTML string to prevent cross-site-scripting (XSS) vulnerabilities.
  - Removes characters and constructs that can trick browsers.
  - Makes sure all HTML entities are well-formed.
  - Makes sure all HTML tags and attributes are well-formed.
  - Makes sure no HTML tags contain URLs with a disallowed protocol (e.g. javascript:).

filter_xss_admin($string)

- Very permissive XSS/HTML filter for admin-only use.
- Use only for fields where it is impractical to use the whole filter system, but where some (mainly inline) mark-up is desired (so check_plain() is not acceptable).
- Allows all tags that can be used inside an HTML body, save for scripts and styles.

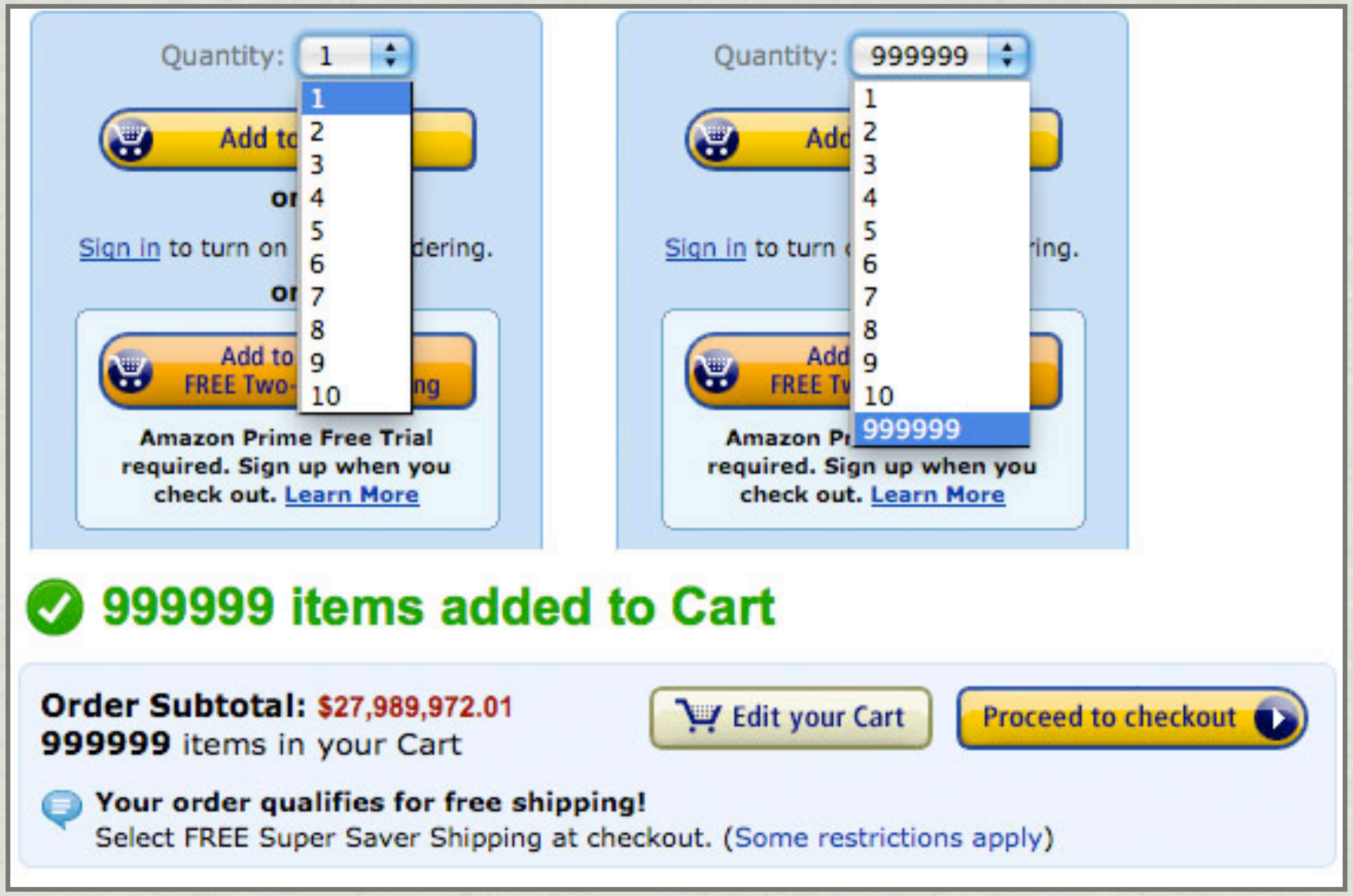
Sending Email

- drupal_mail($module, $key, $to, $language, $params = array(), $from = NULL, $send = TRUE);
- Sending an e-mail works with defining an e-mail template (subject, text and possibly e-mail headers) and the replacement values to use in the appropriate places in the template.
  - Doesn’t allow headers to be injected.
    - Users cannot add a Bcc via the subject line

drupalize.me

Drupal Forms

- Drupal Forms API (FAPI) protects against XSRF using a token and session system that checks for validity of POST data.



Amazon Prime Free Trial required. Sign up when you check out. Learn More

999999 items added to Cart

Order Subtotal: $27,989,972.01
999999 items in your Cart

Your order qualifies for free shipping!
Select FREE Super Saver Shipping at checkout. (Some restrictions apply)

drupalize.me

SQL Injection

- The query builder with variable replacement uses the database API to safely handle the data

-
  ```
  db_merge('example')
    ->key(array('name' => $name))
    ->fields(array(
        'field1' => $value1,
        'field2' => $value2,))
    ->execute();
  ```

HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.
http://xkcd.com/327

drupalize.me

Wash All User-submitted Data On Output

- Don’t trust what people enter in your site.
- Use the tools that Drupal provides to protect yourself against vulnerabilities.
- Documentation
  http://drupal.org/writing-secure-code
- Sanitization functions
  http://api.drupal.org/api/drupal/includes--common.inc/group/sanitization/7