How do you implement a good profanity filter?

Asked 16 years, 1 month ago Modified 3 years, 5 months ago

Viewed 153k times Part of PHP Collective

228

Many of us need to deal with user input, search queries, and situations where the input text can potentially contain profanity or undesirable language. Oftentimes this needs to be filtered out.

Where can one find a good list of swear words in various languages and dialects?

Are there APIs available to sources that contain good lists? Or maybe an API that simply says "yes this is clean" or "no this is dirty" with some parameters?

What are some good methods for catching folks trying to trick the system, like a$$, azz, or a55?

Bonus points if you offer solutions for PHP. :)

Edit: Response to answers that say simply avoid the programmatic issue:

I think there is a place for this kind of filter when, for instance, a user can use public image search to find pictures that get added to a sensitive community pool. If they can search for "penis", then they will likely get many pictures of, yep. If we don't want pictures of that, then preventing the word as a search term is a good gatekeeper, though admittedly not a foolproof method. Getting the list of words in the first place is the real question.

So I'm really referring to a way to figure out of a single token is dirty or not and then simply disallow it. I'd not bother preventing a sentiment like the totally hilarious "long necked giraffe" reference. Nothing you can do there. :)

PHP php regex user-input

Share Follow

edited Dec 1, 2019 at 23:46
S.S. Anne
15.5k ● 8 ● 41 ● 81

asked Nov 7, 2008 at 20:19
Ben Throop
4,833 ● 5 ● 25 ● 20

15 It's a shame that all the top answers are existential and defeatist diversions from the programming challenge. With "cyborg" computational services like Mechanical Turk gaining steam, and almost all software becoming social, it is more important than ever to have a heuristic to red-flag content and bring it to the attention of a moderator! – JasonSmith Sep 29, 2009 at 11:02

12 Please be careful about the language context, especially if you're doing i18n. I once tried to set up a Google Group for the course I was giving called "Sanal ortamda görselleştirme" which is turkish for "Visualization in virtual media". Google was stupid enough to reject it *because the title contained the word "anal"*. Sanal[tr]=Virtual[en] and Google shamelessly accused me of profanity! :D Please don't let weird things like this happen. – edgerunner Sep 19, 2010 at 12:05

What about if you seach for the word in *Spanish*? You can actually get around Google Images filter that way (if you are localized to some other language). – new123456 Jul 21, 2011 at 1:46

Another suggestion would be NOT to prohibit these words, but to register the users that use them. If an user/IP gets more than 2, 3 or whatever you want, then block that person. Not foolproof neither, but I think it's much more inconvenient to be blocked and have to change user/IP/both than to write 'fluffy white bunny' instead of 'pussy'. A part, users don't know WHAT words or expressions they cannot use, so they cannot just guess different bad words so easily as they get banned. – Francisco Presencia Jul 15, 2012 at 8:19

2 Profanity filters are a bad idea. It's very hard to differentiate between someone trying to trick the system ("Fudge you!") and someone legitimately talking about something totally appropriate ("I like chocolate fudge.") – clickbait Jun 3, 2016 at 20:05

19 Answers

Sorted by: Highest score (default) ⇕

186

[Obscenity Filters: Bad Idea, or Incredibly Intercoursing Bad Idea?]

Also, one can't forget The Untold History of Toontown's SpeedChat, where even using a "safe-word whitelist" resulted in a 14-year-old quickly circumventing it with: *"I want to stick my long-necked Giraffe up your fluffy white bunny."*

Bottom line: Ultimately, for any system that you implement, there is absolutely no substitute for human review (whether peer or otherwise). Feel free to implement a rudimentary tool to get rid of the drive-by's, but for the determined troll, you absolutely must have a non-algorithm-based approach.

A system that removes anonymity and introduces accountability (something that Stack Overflow does well) is helpful also, particularly in order to help combat John Gabriel's G.I.F.T.

You also asked where you can get profanity lists to get you started -- one open-source project to check out is Dansguardian -- check out the source code for their default profanity lists. There is also an additional third party Phrase List that you can download for the proxy that may be a helpful gleaning point for you.

Edit in response to the question edit: Thanks for the clarification on what you're trying to do. In that case, if you're just trying to do a simple word filter, there are two ways you can do it. One is to create a single long regexp with all of the banned phrases that you want to censor, and merely do a regex find/replace with it. A regex like:

```
$filterRegex = "(boogers|snot|poop|shucks|argh)"
```

and run it on your input string using preg_match() to wholesale test for a hit,

or preg_replace() to blank them out.

You can also load those functions up with arrays rather than a single long regex, and for long word lists, it may be more manageable. See the preg_replace() for some good examples as to how arrays can be used flexibly.

For additional PHP programming examples, see this page for a somewhat advanced generic class for word filtering that *'s out the center letters from censored words, and this previous Stack Overflow question that also has a PHP example (the main valuable part in there is the SQL-based filtered word approach -- the leet-speak compensator can be dispensed with if you find it unnecessary).

You also added: "*Getting the list of words in the first place is the real question.*" -- in addition to some of the previous Dansgaurdian links, you may find this handy .zip of 458 words to be helpful.

Share Follow

edited Jul 8, 2021 at 13:48

community wiki

12 revs, 6 users 89%
HanClinto

@JPLemme: Yes it should -- I should have added [sic] afterwards, since that's how Atwood spelled it. :) – HanClinto Nov 11, 2008 at 18:37

"Club Penguin" adds hundreds of entries to their profanity filter *every day*: raphkoster.com/2008/05/09/… – Frank Farmer Jun 20, 2009 at 0:02

7 A word boundary wrapper around your regex options would prevent the **clbuttic** mistake – cjk Apr 27, 2010 at 13:03

@ck: Only if you're not worried about being able to filter out mis-spelled words "F*ckkkk yo' asssss" :) I'm not sure I trust my trolls to have very precise spelling. – HanClinto Apr 27, 2010 at 20:46

1 If you'd like to hear the "Fluffy White Bunny" story from the proverbial horse's mouth, it's a podcast episode now: socialmediaclarity.tumblr.com/post/70499341079/… – F. Randall Farmer Dec 20, 2013 at 8:39

53

Whilst I know that this question is fairly old, but it's a commonly occurring question...

There is both a reason and a distinct need for profanity filters (see Wikipedia entry here), but they often fall short of being 100% accurate for very distinct reasons; **Context** and **accuracy**.

It depends (wholly) on what you're trying to achieve - at it's most basic, you're probably trying to cover the "seven dirty words" and then some... Some businesses need to filter the most basic of profanity: basic swear words, URLs or even personal information and so on, but others need to prevent illicit account naming (Xbox live is an example) or far more...

User generated content doesn't just contain potential swear words, it can also contain offensive references to:

- Sexual acts
- Sexual orientation
- Religion
- Ethnicity
- Etc...

And potentially, in multiple languages. Shutterstock has developed basic dirty-words lists in 10 languages to date, but it's still basic and very much oriented towards their 'tagging' needs. There are a number of other lists available on the web.

I agree with the accepted answer that it's not a defined science and *as* language is a continually evolving *challenge* but one where a 90% catch rate is better than 0%. It depends purely on your goals - what you're trying to achieve, the level of support you have and how important it is to remove profanities of different types.

In building a filter, you need to consider the following elements and how they relate to your project:

- Words/phrases

- Acronyms (FOAD/LMFAO etc)
- False positives (words, places and names like 'mishit', 'scunthorpe' and 'titsworth')
- URLs (porn sites are an obvious target)
- Personal information (email, address, phone etc - if applicable)
- Language choice (usually English by default)
- Moderation (how, if at all, you can interact with user generated content and what you can do with it)

You can easily build a profanity filter that captures 90%+ of profanities, but you'll never hit 100%. It's just not possible. The closer you want to get to 100%, the harder it becomes... Having built a complex profanity engine in the past that dealt with more than 500K realtime messages per day, I'd offer the following advice:

A basic filter would involve:

- Building a list of applicable profanities
- Developing a method of dealing with derivations of profanities

A moderately complex filer would involve, (In addition to a basic filter):

- Using complex pattern matching to deal with extended derivations (using advanced regex)
- Dealing with Leetspeak (l33t)

- Dealing with false positives

A complex filter would involve a number of the following (In addition to a moderate filter):

- Whitelists and blacklists
- Naive bayesian inference filtering of phrases/terms
- Soundex functions (where a word sounds like another)
- Levenshtein distance
- Stemming
- Human moderators to help guide a filtering engine to learn by example or where matches aren't accurate enough without guidance (a self/continually-improving system)
- Perhaps some form of AI engine

Share Follow

edited Nov 19, 2012 at 5:42

answered Nov 19, 2012 at 4:27

nickhar

20.7k ● 12 ● 63 ● 77

30

I don't know of any good libraries for this, but whatever you do, make sure that you err in the direction of letting stuff through. I've dealt with systems that wouldn't allow

▼

me to use "mpassell" as a username, because it contains "ass" as a substring. That's a great way to alienate users!

Share Follow

answered Nov 7, 2008 at 20:26

Matt Passell

4,749 ● 3 ● 26 ● 39

23 or forbidding "cockpit" in a flying spaceships game – Shinhan Nov 7, 2008 at 21:28

▲

27

▼

a profanity filtering system will never be perfect, even if the programmer is cocksure and keeps abreast of all nude developments

that said, any list of 'naughty words' is likely to perform as well as any other list, since the underlying problem is *language understanding* which is pretty much intractable with current technology

so, the only practical solution is twofold:

1. be prepared to update your dictionary frequently
2. hire a human editor to correct false positives (e.g. "clbuttic" instead of "classic") and false negatives (oops! missed one!)

Share Follow

edited Aug 24, 2009 at 20:49

answered Nov 7, 2008 at 22:27

Steven A. Lowe
61.1k ● 19 ● 135 ● 204

1 Just detect the word with spaces either side, full stop after, No? – David D Nov 21, 2013 at 17:37

2 H3ll no man, that only works for the most trivial of cases; we're dealing with humans here, and they're quite clever :) – Steven A. Lowe Nov 21, 2013 at 17:53

▲

26

▼

During a job interview of mine, the company CTO who was interviewing me tried out a word/web game I wrote in Java. Out of a word list of the entire Oxford English dictionary, what was the first word that came up to be guessed?

Of course, the most foul word in the English language.

Somehow, I still got the job offer, but I then tracked down a profanity word list (not unlike this one) and wrote a quick script to generate a new dictionary without all of the bad words (without even having to look at the list).

For your particular case, I think comparing the search to real words sounds like the way to go with a word list like that. The alternative styles/punctuation require a bit more work, but I doubt users will use that often enough to be an issue.

Share Follow

answered Nov 7, 2008 at 22:36

Matthew

1,155 ● 1 ● 9 ● 8

8 Off topic, but what is the most foul word? I always considered it the c word or the n word, but I assume that people think the f word is – Jeff Oct 6, 2011 at 23:29

2 *"I doubt users will use that often enough to be an issue"* Keep hope alive. Once users stumble on the filter, they will work on finding ways of circumvention. It could be as simple as replacing letters with numbers to odd placement of spaces, etc). – BryanH Nov 16, 2012 at 3:26

14

The only way to prevent offensive user input is to prevent all user input.

If you insist on allowing user input and need moderation, then incorporate human moderators.

Share Follow

answered Nov 7, 2008 at 20:42

Axel

213 ● 1 ● 2 ● 7

9

I collected 2200 bad words in 12 languages: en, ar, cs, da, de, eo, es, fa, fi, fr, hi, hu, it, ja, ko, nl, no, pl, pt, ru, sv, th, tlh, tr, zh.

MySQL dump, JSON, XML or CSV options are available.

https://github.com/turalus/openDB

I'd suggest you to execute this SQL into your DB and check everytime when user inputs something.

Share Follow

answered Mar 3, 2017 at 8:22

Tural Ali
23.2k ● 18 ● 81 ● 131

8

Beware of localization issues: what is a swearword in one language might be a perfectly normal word in another.

One current example of this: ebay uses a dictionary approach to filter "bad words" from feedback. If you try to enter the german translation of "this was a perfect transaction" ("das war eine perfekte Transaktion"), ebay will reject the feedback due to bad words.

Why? Because the german word for "was" is "war", and "war" is in ebay dictionary of "bad words".

So beware of localisation issues.

Share Follow

answered Apr 27, 2010 at 12:55

Sam
29k ● 50 ● 170 ● 250

7

Regarding your "trick the system" subquestion, you can handle that by normalizing both the "bad word" list and the user-entered text before doing your search. e.g., Use a series of regexes (or **tr** if PHP has it) to convert **[z$5]** to "s", **[4@]** to "a", etc., then compare the normalized "bad word" list against the normalized text. Note that the normalization could potentially lead to additional false positives, although I can't think of any actual cases at the moment.

The larger challenge is to come up with something that will let people quote "The **pen is** mightier than the sword" while blocking "p e n i s".

Share Follow

answered Nov 8, 2008 at 1:35

Dave Sherohman
46.1k ● 14 ● 66 ● 103

14 Don't forget experts-exchange.com and pen-island.com; those site urls once didn't contain hyphens. – BryanH Feb 5, 2013 at 19:10

6

If you can do something like Digg/Stackoverflow where the users can downvote/mark obscene content... do so.

Then all you need to do is review the "naughty" users, and block them if they break the rules.

Share Follow

answered Nov 7, 2008 at 20:46

scunliffe
63.5k ● 26 ● 131 ● 165

I'm a little late to the party, but I have a solution that might work for some who read this. It's in javascript instead of php, but there's a valid reason for it.

> Full disclosure, I wrote this plugin...

Anyways.

The approach I've gone with is to allow a user to "Opt-In" to their profanity filtering. Basically profanity will be allowed by default, but if my users don't want to read it, they don't have to. This also helps with the "l33t sp3@k" issue.

The concept is a simple `jquery` plugin that gets injected by the server if the client's account is enabling profanity filtering. From there, it's just a couple simple lines that blot out the swears.

Here's the demo page
https://chaseflorell.github.io/jQuery.ProfanityFilter/demo/

```
<div id="foo">
    ass will fail but password will not
</div>

<script>
    // code:
    $('#foo').profanityFilter({
        customSwears: ['ass']
    });
</script>
```

result

> *** will fail but password will not

Share Follow

edited Aug 19, 2015 at 20:01

answered Oct 29, 2012 at 3:22

Chase Florell
47.3k ● 59 ● 190 ● 382

Here's a [jsFiddle working demo](#) to accompany this answer.
– Chase Florell Aug 19, 2015 at 19:55 ✎

Very naive. Didn't filter `a$$` – wingerse Apr 8, 2016 at 21:18

3 @EmperorAiman it was never intended to filter out [l33t speak](#). I don't recommend trying to filter that, as it's a losing battle. The profanity filter that I've got posted is "built in order to allow users to "Opt-in" to profanity filtering". meaning that it's best used on a site that allows profanity by default. If you want to filter `a$$`, then you add it to the filter list.
– Chase Florell Apr 12, 2016 at 17:37

▲

4

▼

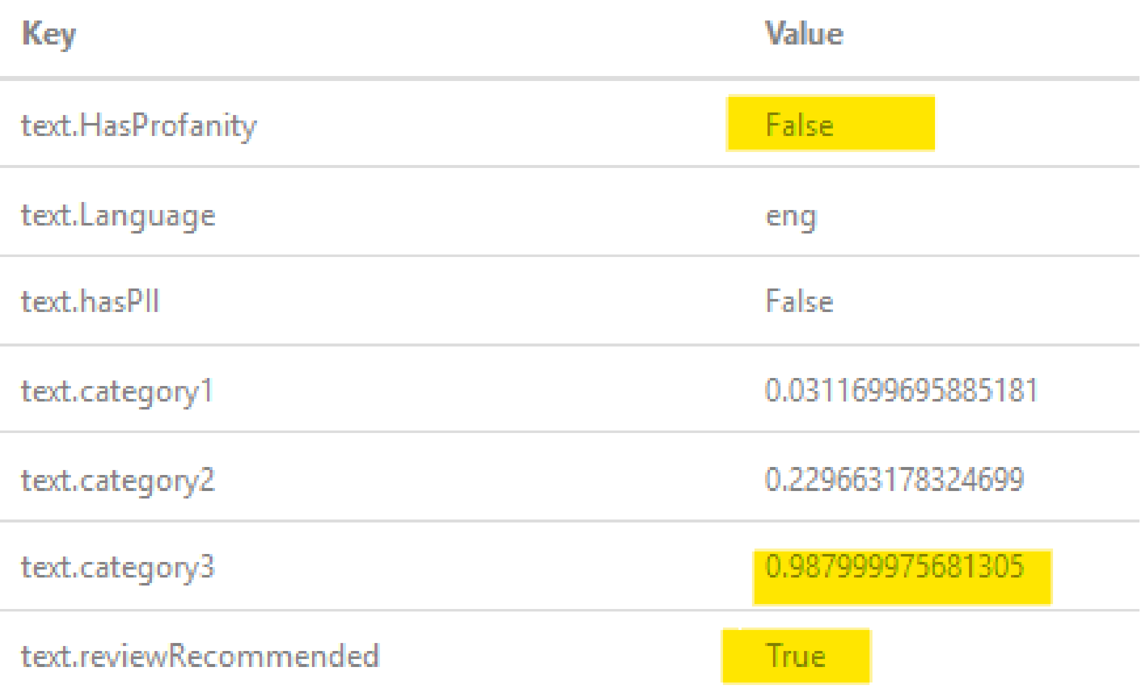
Also late in the game, but doing some researches and stumbled across here. As others have mentioned, it's just almost close to impossible if it was automated, but if your design/requirement can involve in some cases (but not all the time) human interactions to review whether it is profane or not, you may consider ML.
https://learn.microsoft.com/en-us/azure/cognitive-

services/content-moderator/text-moderation-api#profanity is my current choice right now for multiple reasons:

- Supports many localization
- They keep updating the database, so I don't have to keep up with latest slangs or languages (maintenance issue)
- When there is a high probability (I.e. 90% or more) you can just deny it pragmatically
- You can observe for category which causes a flag that may or may not be profanity, and can have somebody review it to teach that it is or isn't profane.

For my need, it was/is based on public-friendly commercial service (OK, videogames) which other users may/will see the username, but the design requires that it has to go through profanity filter to reject offensive username. The sad part about this is the classic "clbuttic" issue will most likely occur since usernames are usually single word (up to N characters) of sometimes multiple words concatenated... Again, Microsoft's cognitive service will not flag "Assist" as Text.HasProfanity=true but may flag one of the categories probability to be high.

As the OP inquires, what about "a$$", here's a result when I passed it through the filter:



Key	Value
text.HasProfanity	False
text.Language	eng
text.hasPII	False
text.category1	0.0311699695885181
text.category2	0.229663178324699
text.category3	0.987999975681305
text.reviewRecommended	True

, as you can see, it has determined it's not profane, but it has high probability that it is, so flags as recommendations of reviewing (human interactions).

When probability is high, I can either return back "I'm sorry, that name is already taken" (even if it isn't) so that it is less offensive to anti-censorship persons or something, if we don't want to integrate human review, or return "Your username have been notified to the live operation department, you may wait for your username to be reviewed and approved or chose another username". Or whatever...

By the way, the cost/price for this service is quite low for my purpose (how often does the username gets changed?), but again, for OP maybe the design demands more intensive queries and may not be ideal to pay/subscribe for ML-services, or cannot have human-review/interactions. It all depends on the design... But if design does fit the bill, perhaps this can be OP's solution.

If interested, I can list the cons in the comment in the future.

Share Follow

answered Apr 25, 2019 at 16:47

HidekiAI

3,053 ● 3 ● 21 ● 23

3

Once you have a good MYSQL table of some bad words you want to filter (I started with one of the links in this thread), you can do something like this:

```
$errors = array();  //Initialize error array (I use th
validations)

$SCREENNAME = mysql_real_escape_string($_POST['SCREENN
input data to prevent SQL injection when you query the

$ProfanityCheckString = strtoupper($SCREENNAME); //Mak
uppercase (so that 'BaDwOrD' is the same as 'BADWORD')
profanity table will need to be UPPERCASE for this to

$ProfanityCheckString = preg_replace('/[_-]/','',$Prof
allow alphanumeric, underscores, and dashes...nothing
PHP form validation).  Pull out non-alphanumeric chara
shows up as 'BADWORD'.

$ProfanityCheckString = preg_replace('/1/','I',$Profan
//Replace common numeric representations of letters so
'BADWORD'.

$ProfanityCheckString = preg_replace('/3/','E',$Profan

$ProfanityCheckString = preg_replace('/4/','A',$Profan

$ProfanityCheckString = preg_replace('/5/','S',$Profan

$ProfanityCheckString = preg_replace('/6/','G',$Profan
$ProfanityCheckString = preg_replace('/7/','T',$Profan

$ProfanityCheckString = preg_replace('/8/','B',$Profan

$ProfanityCheckString = preg_replace('/0/','O',$Profan
//Replace ZERO's with O's (Capital letter o's).

$ProfanityCheckString = preg_replace('/Z/','S',$Profan
//Replace Z's with S's, another common substitution.
with S's in your profanity database for this to work p
the numbers too--having S3X7 in your database won't wo
render that string as 'SEXY'.  The profanity table sho
version of the bad words.

$CheckProfanity = mysql_query("SELECT * FROM DATABASE.
'".$ProfanityCheckString."'");
if(mysql_num_rows($CheckProfanity) > 0) {$errors[] = '
Screen Name.';} //Check your profanity table for the s
get real crazy using LIKE and wildcards, but I only wa
filter.

if (count($errors) > 0) {foreach($errors as $error) {$
class='PHPError'>$error</span><br /><br />";} echo $er
PHP errors that come out of the validation, including


//You can also use these lines to troubleshoot.
//echo $ProfanityCheckString;
//echo "<br />";
//echo mysql_error();
//echo "<br />";
```

I'm sure there is a more efficient way to do all those replacements, but I'm not smart enough to figure it out (and this seems to work okay, albeit inefficiently).

I believe that you should err on the side of allowing users to register, and use humans to filter and add to your profanity table as required. Though it all depends on the cost of a false positive (okay word flagged as bad) versus a false negative (bad word gets through). That should ultimately govern how aggressive or conservative you are in your filtering strategy.

I would also be very careful if you want to use wildcards, since they can sometimes behave more onerously than you intend.

Share Follow

answered Aug 16, 2011 at 2:33

andrew

31 ● 1

2

I agree with HanClinto's post higher up in this discussion. I generally use regular expressions to string-match input text. And this is a vain effort, as, like you originally mentioned you have to explicitly account for every trick form of writing popular on the net in your "blocked" list.

On a side note, while others are debating the ethics of censorship, I must agree that some form is necessary on the web. Some people simply enjoy posting vulgarity because it can be instantly offensive to a large body of people, and requires absolutely no thought on the author's part.

Thank you for the ideas.

HanClinto rules!

Share Follow

answered Feb 24, 2009 at 20:30

msm516

1

Frankly, I'd let them get the "trick the system" words out and ban them instead, which is just me. But it also makes the programming simpler.

What I'd do is implement a regex filter like so: `/[\s]dooby (doo?)[\s]/i` or it the word is prefixed on others, `/[\s]doob(er|ed|est)[\s]/`. These would prevent filtering words like assuaged, which is perfectly valid, but would also require knowledge of the other variants and updating the actual filter if you learn a new one. Obviously these are all examples, but you'd have to decide how to do it yourself.

I'm not about to type out all the words I know, not when I don't actually want to know them.

Share Follow

answered Nov 7, 2008 at 20:25

Robert K

30.3k ● 12 ● 63 ● 79

1

Don't. It just leads to problems. One clbuttic personal experience I have with profanity filters is the time where I was kick/banned from an IRC channel for mentioning that I was "heading over the bridge to Hancock for a couple hours" or something to that effect.

Share Follow

answered Nov 7, 2008 at 20:37

Adam Jaskiewicz

11k ● 3 ● 36 ● 37

1

I agree with the futility of the subject, but if you have to have a filter, check out Ning's Boxwood:

> Boxwood is a PHP extension for fast replacement of multiple words in a piece of text. It supports case-sensitive and case-insensitive matching. It requires that the text it operates on be encoded as UTF-8.

Also see this blog post for more details:

- Fast Multiple String Replacement in PHP

> With Boxwood, you can have your list of search terms be as long as you like -- the search and replace algorithm doesn't get slower with more words on the list of words to look for. It works by building a trie of all the search terms and then scans your subject text just once, walking down elements of the trie and comparing them to characters in your text. It supports US-ASCII and UTF-8, case-sensitive or insensitive matching, and has some English-centric word boundary checking logic.

Share Follow

answered Sep 30, 2010 at 9:01

Gordon

317k ● 76 ● 545 ● 565

▲

1

▼

I concluded, in order to create a good profanity filter we need 3 main components, or at least it is what I am going to do. These they are:

1. The filter: a background service that verify against a blacklist, dictionary or something like that.
2. Not allow anonymous account
3. Report abuse

A bonus, it will be to reward somehow those who contribute with accurate abuse reporters and punish the offender, e.g. suspend their accounts.

Share Follow

edited Apr 22, 2015 at 14:27

answered Oct 12, 2012 at 2:51

Jaider

14.9k ● 5 ● 80 ● 85

-10

Don't.

Because:

- Clbuttic
- Profanity is not OMG EVIL
- Profanity cannot be effectively defined
- Most people quite probably don't appreciate being "protected" from profanity

Edit: While I agree with the commenter who said "censorship is wrong", that is not the nature of this answer.

Share Follow

edited Nov 16, 2012 at 3:20

BryanH
6,042 ● 3 ● 36 ● 47

answered Nov 7, 2008 at 20:22

eyelidlessness
63.5k ● 12 ● 92 ● 95

96 10 upvotes for this non-answer? As if anybody who wants to filter profanity must be a moralizing half wit? Good grief. This is a valid question and snarky drive-by responses shouldn't be rewarded. -1. – Kluge Nov 7, 2008 at 21:45

12 @Kludge: You're the only one who said "moralizing half wit", in fact I said nothing about the moral nature of implementing a profanity filter at all. Mitch brings up part of the reason I said "don't", and it's not a snarky drive-by. Sometimes "don't" is the correct answer to "how do I...?" [cont'd]
– eyelidlessness Nov 8, 2008 at 1:57

2 @eyelidlessness: Perhaps you are right that I read too much into your single-word answer. But since you didn't elaborate, I couldn't tell if your objections were on moral grounds or technical ones. I'll admit that I'm tired of "censorship in any form is bad" comments. – Kluge Nov 8, 2008 at 17:09

5 -1. "Don't" is not a valid answer regardless of moral or technical issues. There are plenty of times where it is perfectly appropriate to filter content based on the nature of the content. Imagine an ecommerce site selling women's undergarments and offering a 'Reviews' feature. Do you really want prepubescent boys littering your site with garbage? Of course not. And maybe it's too cumbersome to have a human approval process. A simple filter to deny reviews with garbage is a good thing. – pspahn Apr 26, 2012 at 20:32

3 @pspahn, "don't" is the correct answer to any question asking for a solution to the wrong problem. There are certainly valid cases where content should be moderated, but a "profanity filter" is not it. – eyelidlessness Apr 26, 2012 at 22:03

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.