The Seven Righteous Fights

You Should Be Fighting
Heidi Waterhouse
LISA 2016

#7fights
h.waterhouse@gmail.com
@wiredferret
Definition of a Technical Writer:

A master of assembling technical clues into a narrative whole
This truck is stuck in the mud

@wiredferret
Example
Tower of translation Babel

@wiredferret

#7fights
5 days

Plus two weeks
Fingers crossed

@wiredferret

#7fights
You cannot fix it later
The Seven Righteous Fights

1. Localization
2. Security
3. Extensibility
4. Documentation
5. Affordance
6. Acceptance
7. Accessibility
Localization
Localisation
Localization

Don’t hard code interface elements

No words in logos or images

Bake in extended character support
Security
The most successful hackers actually look like auditors
Security is neither cheap nor easy,

But it beats the alternative
Encryption
Stop saying that word!
Extensibility
Documentation is not state secrets

Documentation is subtle self promotion
User Documentation

Developer onboarding

@wiredferret #7fights
Production scripts and build sequence

@wiredferret #7fights
Secretive build engineers are bad build engineers
Affordance
Some people have more privileges

@wiredferret
Homunculus of sensation
Acceptance
Not all Legos have the same job
Shhhhhh

Explain nothing
If you can't hire an expert, become a student.
Accessibility
8% of men are colorblind
it's no use, d00d ...

... reception still sucks.
TAB

Temporarily Able-Bodied
Now what?
Tactics

Do what you can

Start where you are.
Use what you have.
Do what you can.

- Arthur Ashe
Tactic 1
Write a coding style guide and follow it.
Tactic 2

Pair programming for best practices.
Tactic 3

Host brown bag lunches.
Tactic 4

Ask questions.
Tactic 5

Add tests for accessibility and usability.
Tactic 6

Learn to write ROI documents.
Tactic 7

Cultivate inclusion, diversity, and representation.
Money is the root of all business decisions
The Seven Righteous Fights

1. Localization - Make users comfy
2. Security - Make users safer
3. Extensibility - Make code usable
4. Documentation - Disaster-proof
5. Affordance - Nudge good behavior
6. Acceptance - Test with users
7. Accessibility - Include people
Don’t build in compounding technical debt that you could trivially avoid.
Be productively lazy.
Credits

Copyright ceridwen and licensed for reuse under this Creative Commons Licence

By Dori (Own work) [Public domain], via Wikimedia Commons

By Bill Bertram (Own work) [CC BY-SA 2.5 (http://creativecommons.org/licenses/by-sa/2.5)], via Wikimedia Commons

https://www.flickr.com/photos/dvortygirl/

https://www.flickr.com/photos/nez/

By Jonathan Joseph Bondhus (Own work) [CC BY-SA 3.0 (http://creativecommons.org/licenses/by-sa/3.0)], via Wikimedia Commons

By User:Bryan Derksen (Own work) [Public domain], via Wikimedia Commons

By Ken Teegardin from Boulder, Boulder (Piggy Bank On Pennies) [CC BY-SA 2.0 (http://creativecommons.org/licenses/by-sa/2.0)], via Wikimedia Commons

By Evan-Amos (Own work) [CC BY-SA 3.0 (http://creativecommons.org/licenses/by-sa/3.0)], via Wikimedia Commons

By Lasart75 (Own work) [GFDL (http://www.gnu.org/copyleft/fdl.html) or CC BY-SA 4.0-3.0-2.5-2.0-1.0 (http://creativecommons.org/licenses/by-sa/4.0-3.0-2.5-2.0-1.0)], via Wikimedia Commons

https://www.flickr.com/photos/3333486@N00/

By Johnnjoy (Own work, based on the work of Anders) [Public domain], via Wikimedia Commons

By Jericho [CC BY 3.0 (http://creativecommons.org/licenses/by/3.0)], via Wikimedia Commons

Joshua Reynolds [Public domain], via Wikimedia Commons

@wiredferret #7fights