#NDCOSLO

SHINY OBJECTS ARE COOL BUT SO IS BUILDING PRODUCTS PEOPLE USE

@jennapederson

What is shiny?

Accidental vs. Essential Complexities

Fred Brooks, 1986

Accidental Complexities

Not essential to the problem being solved

Problems engineers create and can fix

languages, tools, processes, techniques, bells and whistles

Essential Complexities

Directly related to the problem being solved

Very little can fix or remove this complexity

deciding what to build, the humans, design and testing, the complexity of hardware/software

What's the problem with shiny?

Doesn't solve the problem

you know, that one our customer has

Has less impact

a focus only on the shiny is not nearly as impactful to the problem our customer has as focusing on the non-shiny

Always starting

but never finishing

And more...

unknown, unsupported, unmaintained, risky, distracting

Why do we choose shiny?

Starting vs. Finishing

starting is orders of magnitude easier than finishing

New is better

obviously, if it's new and improved it's better, right? maybe. maybe not.

We like to learn

and maybe it's just for our resume or to scratch an itch

How do we build products people use?

Stop focusing on the solution and focus on the actual problem

not just the ones we think our customers have or the ones we've manufactured for ourselves

Focus on the customer

get out of the office and talk to them

Focus on the value

add value to someone's life

Code vs No Code

Why write code when you don't need to?

What if you don't fully know what the problem is and then write a bunch of code?

example 1: a React app vs. a Squarespace site

example 2: a React app vs. a Google form + Zapier to send an email or a text

Library vs Roll Your Own

Does the library do what you need to do?

Is it actively maintained?

Would you be the largest user of the library?

Does using the library and having a dependency on it make it more complicated?

example: the Recaptcha gem vs roll your own

example: Circle CI + ClojureScript frontend

AWS All the Things vs. Heroku

Can you do All the Things?

Do you have time to learn All the Things?

Can you afford to hire someone to do All the Things and maintain/support it?

Can you think critically about the security, scalability, and reliability of All the Things?

Do you even need All the Things?

example: startup day 1 vs. Big Mega Corp

How do we innovate?

Use it to our advantage

chase shiny objects but with a goal in mind. have a strategy.

embrace failure without consequences.

let these shiny objects drive us to learn more about our craft, how to solve real world problems with new tech, how to experiment, how to fail, and how to discover what problems we should be solving.

One Last Bit

AN INSPIRATIONAL BIT

The good devs won't just be chasing shiny objects. They will have a fine balance of finding, evaluating, and using new or bleeding edge tech vs. using the tried and true tech.

The End

Thanks!

find me on the tweeter:
@jennapederson

feedback welcome!