Solution Development Pragmatic Approach

DON’T PANIC

we have 59 slides, 45 minutes, ~45 seconds per slide. It should be enough to cover, fast cars, behavioral finance, price theory, broken windows, marginal utility, covariant, coefficiency, irrationality, rock’n’roll, and hopefu

sanırım 50kusur slide ve 45 dakikada - ki slide başına 50 saniyede şu konulardan

hızlı arabalardan, davranışsal finanstan, fiyat teoreminden, kırık camlardan, marjinal faydadan, kovaryanttan, irrasyonaliteden, rocknrolldan, ve umuyorum ki programlamadan bahsedeceğiz.

panik yapmayın

Aybars Badur

Solution Developer & System Administration
Lead at hipolabs

twitter.com/aybarsbadur
github.com/ybrs
aybars.badur@gmail.com

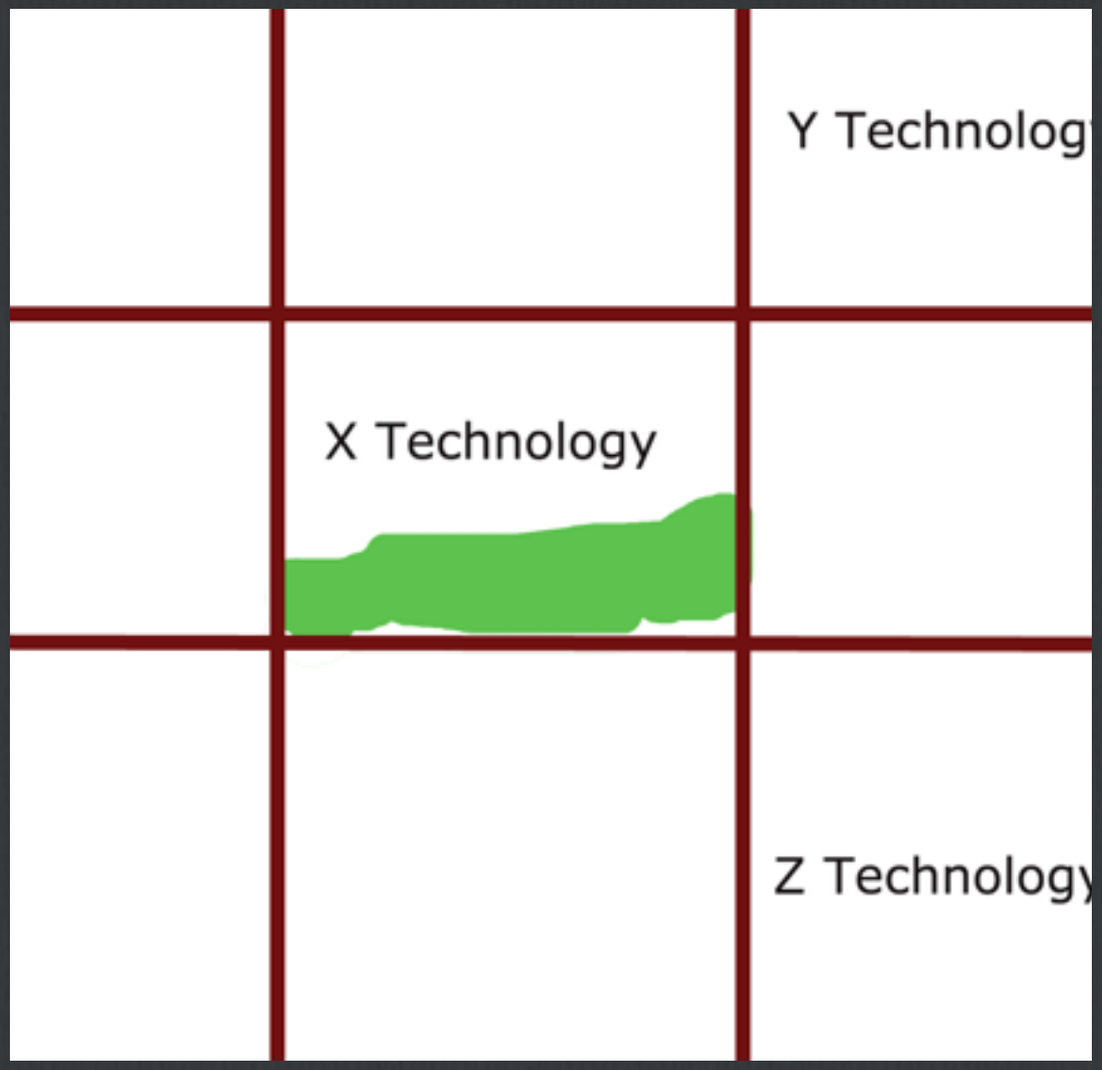
Solution Developer ?

- [ ] Software Developer
- [ ] … Developer
- [ ] Javascript Developer
- [ ] ….

Solution Developer ?

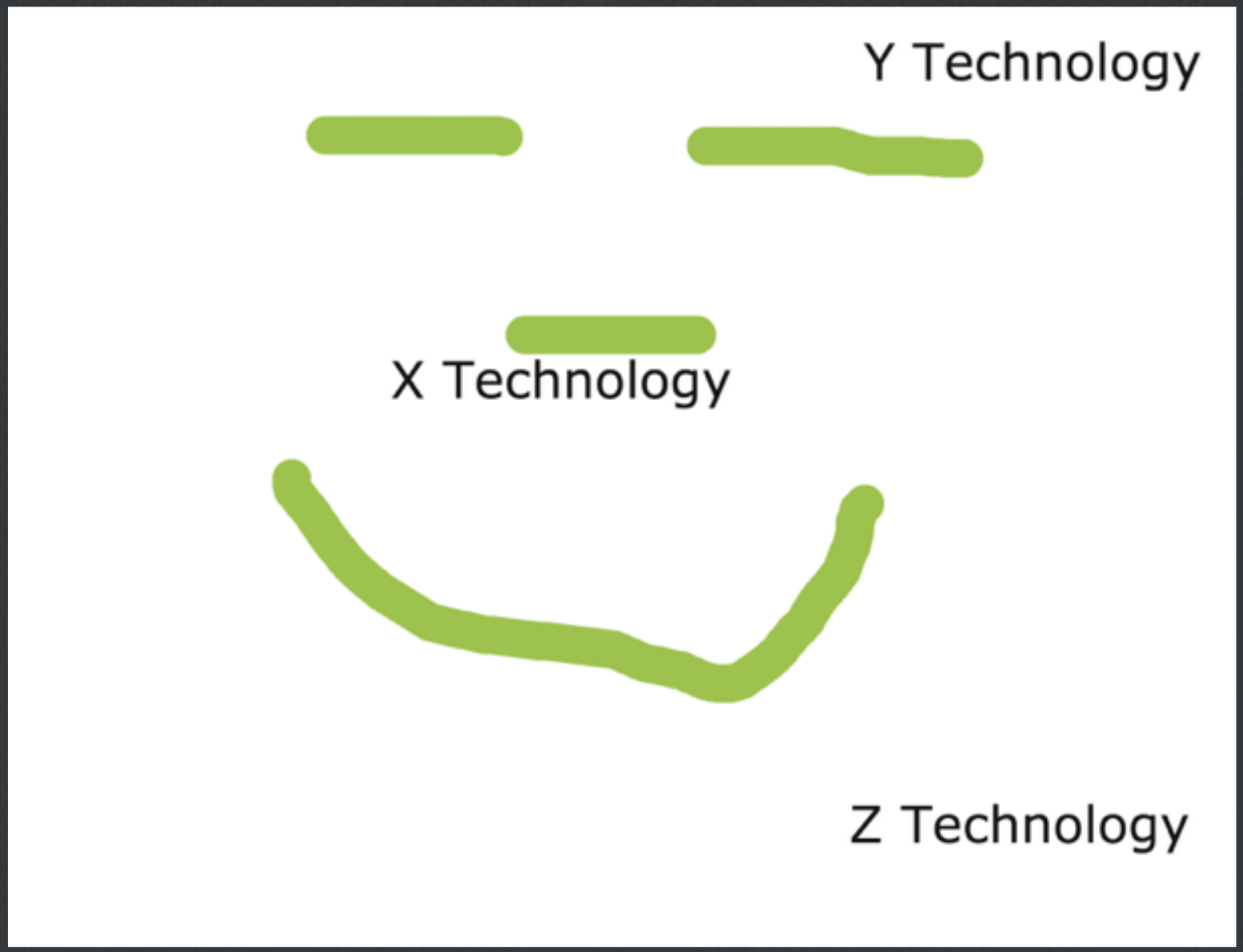
- ☐ Software Developer
  - ☐ Develops Software
- ☐ Java Developer
  - ☐ Develops Java
- ☐ Javascript Developer
  - ☐ Develops Javascript
- ☐ …. Developer
  - ☐ Develops …

Why Have Boundaries ?

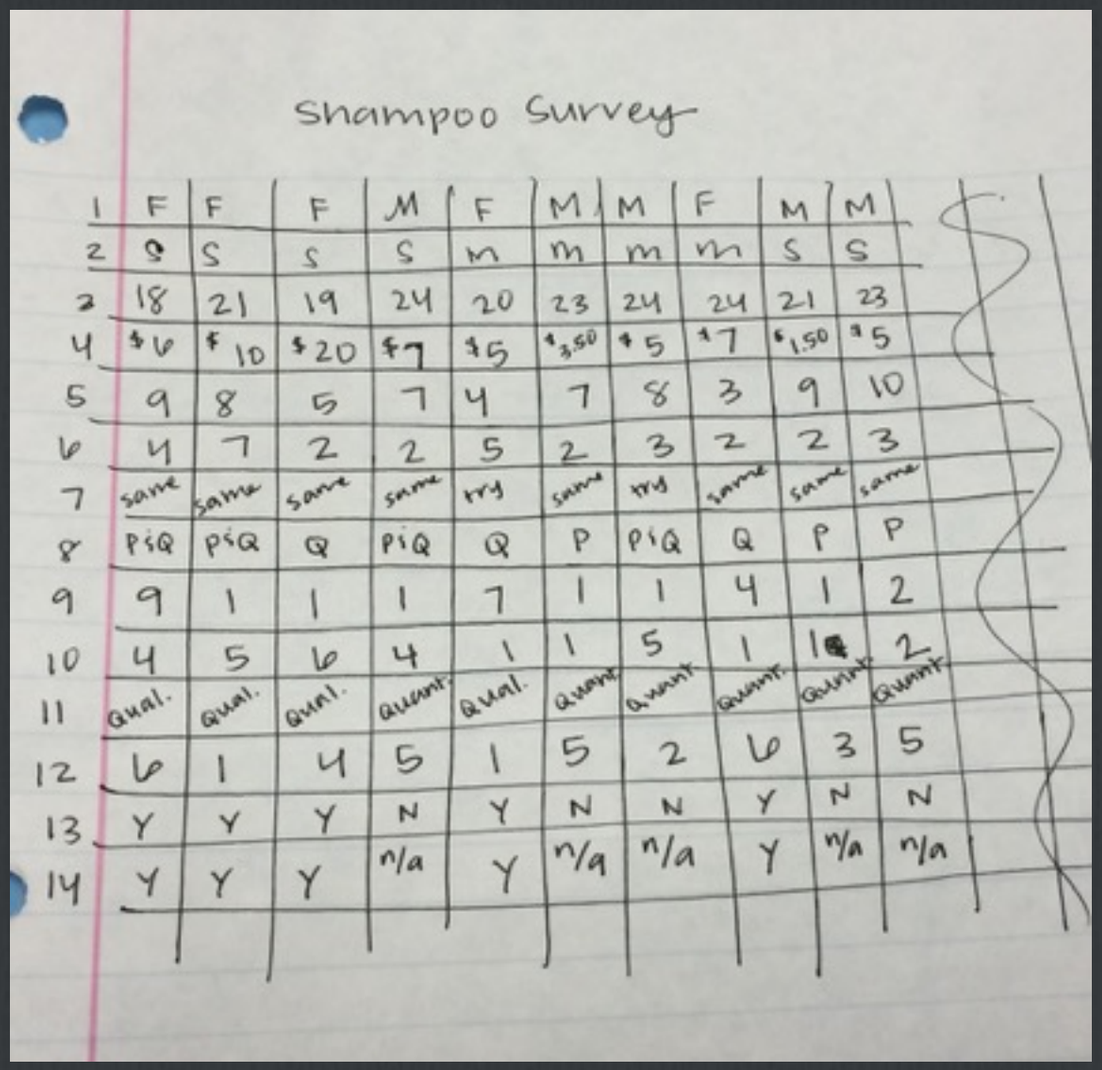


- ☐ X Developer

Why Have Boundaries ?



Developing Solutions

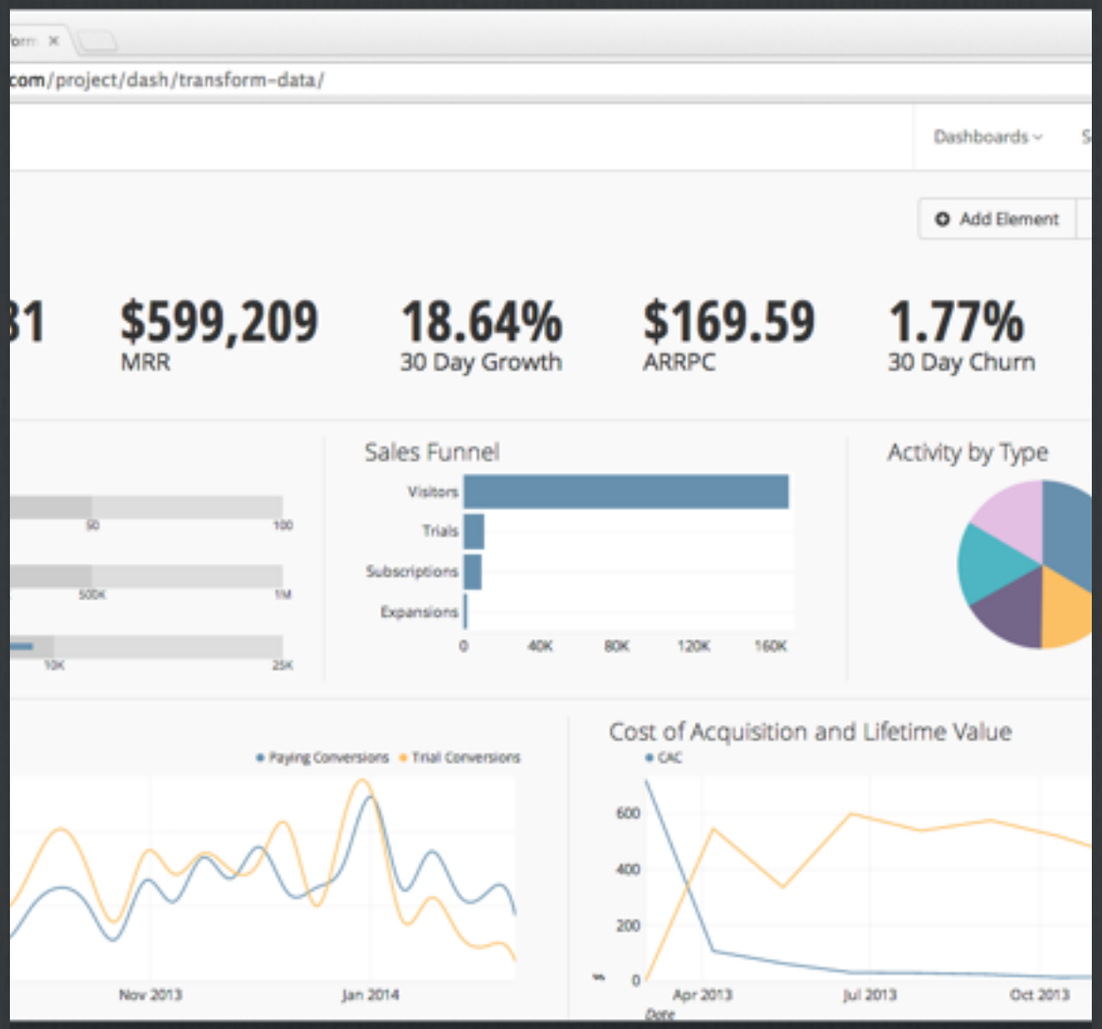


Shampoo Survey

1	F	F	F	M	F	M	M	F	M	M
2	9	S	S	S	M	M	M	M	S	S
3	18	21	19	24	20	23	24	24	21	23
4	$6	$10	$20	$7	$5	$3.50	$5	$7	$1.50	$5
5	9	8	5	7	4	7	8	3	9	10
6	4	7	2	2	5	2	3	2	2	3
7	same	same	same	same	try	same	try	same	same	same
8	P&Q	P&Q	Q	P&Q	Q	P	P&Q	Q	P	P
9	9	1	1	1	7	1	1	4	1	2
10	4	5	6	4	1	1	5	1	1	2
11	Qual.	Qual.	Qual.	Quant.	Qual.	Quant.	Quant.	Quant.	Quant.	Quant.
12	6	1	4	5	1	5	2	6	3	5
13	Y	Y	Y	N	Y	N	N	Y	N	N
14	Y	Y	Y	n/a	Y	n/a	n/a	Y	n/a	n/a

- ☐ **Solutions, doesn't always includes code !**
- ☐ **Data Entry, Form Processing, Questionnaire…**
- ☐ **Maybe Just Use People**

Developing Solutions



- ☐ Solutions, doesn't always includes code !
- ☐ Charts, Dashboards, Admin Interfaces…
- ☐ Look for components/libraries/ open source things/SAAS
- ☐ And buy them

Solution Development

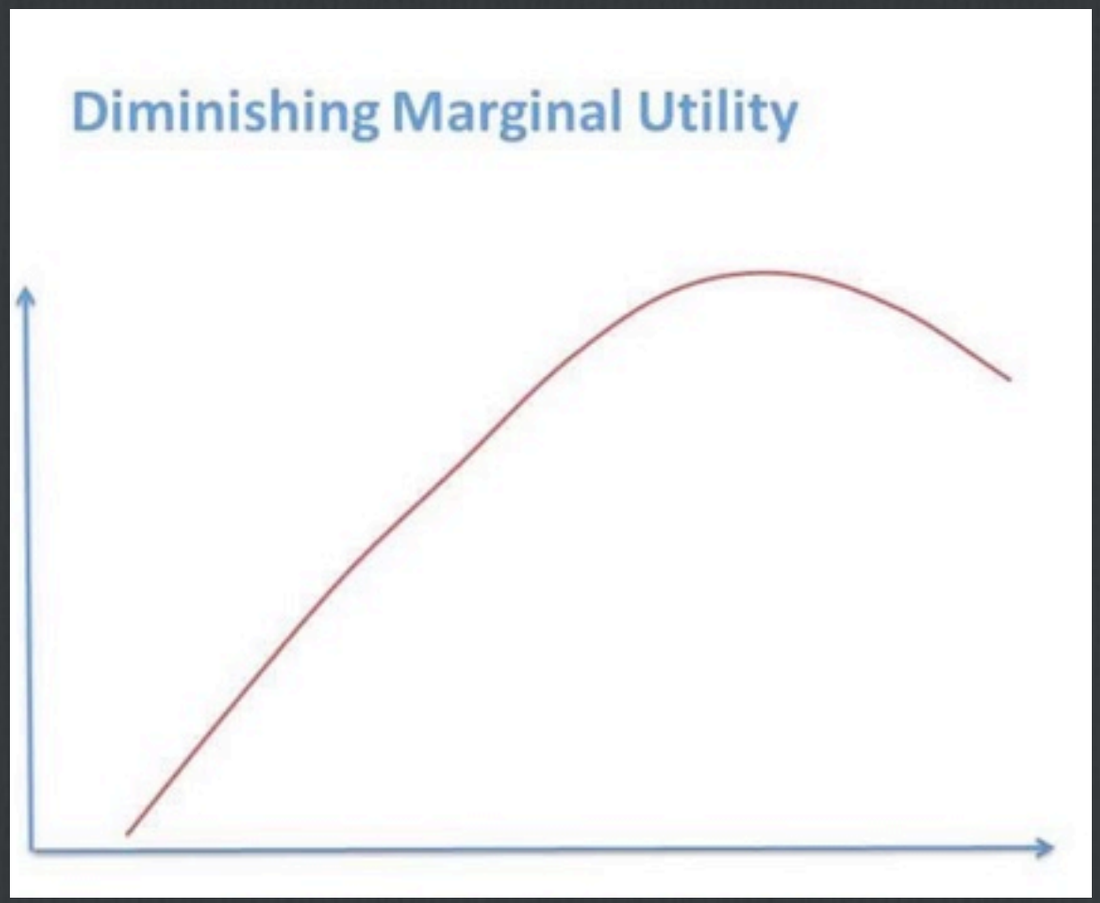
- [ ] Do you love writing code,
- [ ] - OR -
- [ ] creating solutions ?

What is pragmatic anyway ?

“dealing with things sensibly and realistically in a way that is based on practical rather than theoretical considerations.”

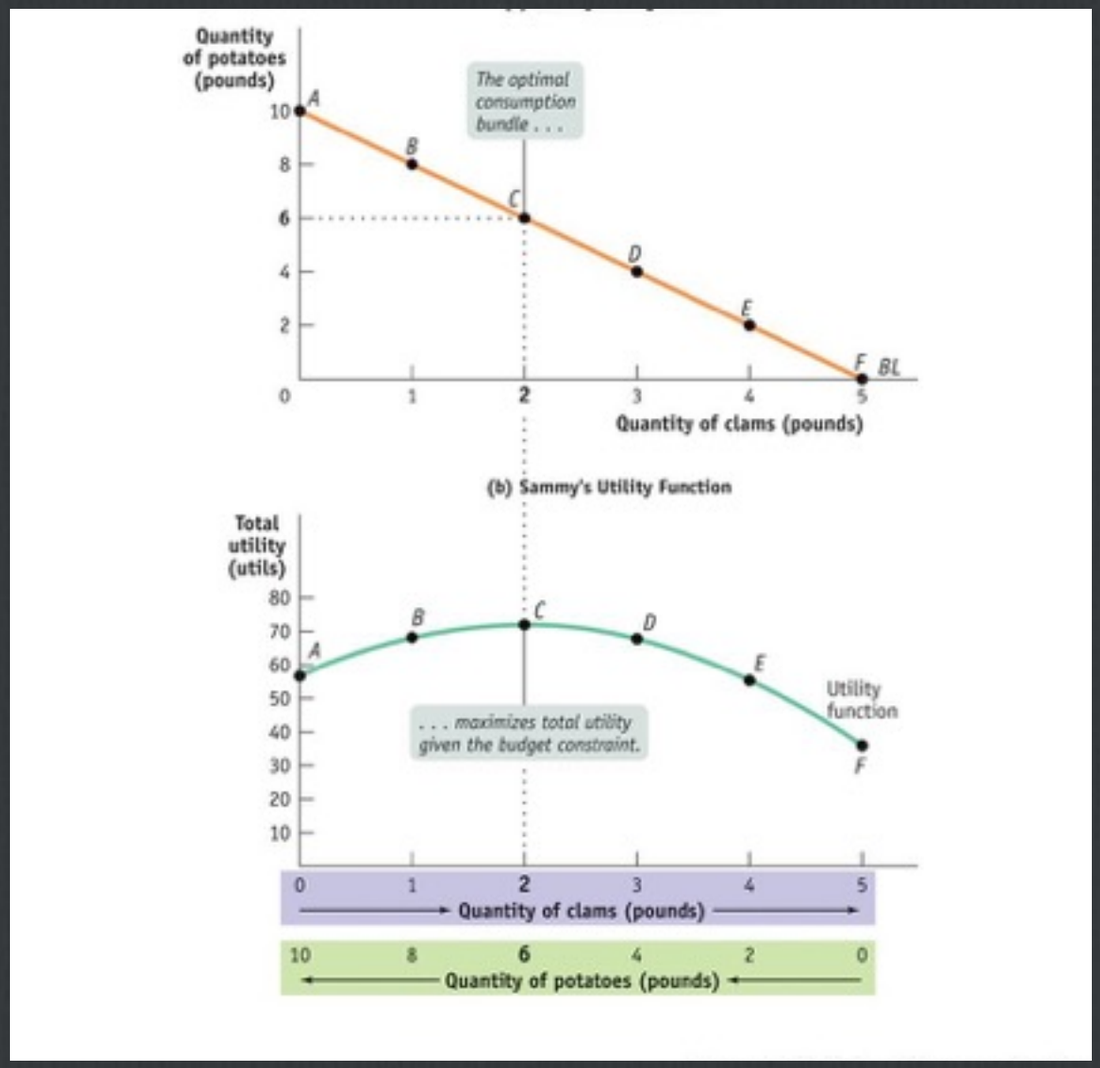
- Google Search

Marginal Utility



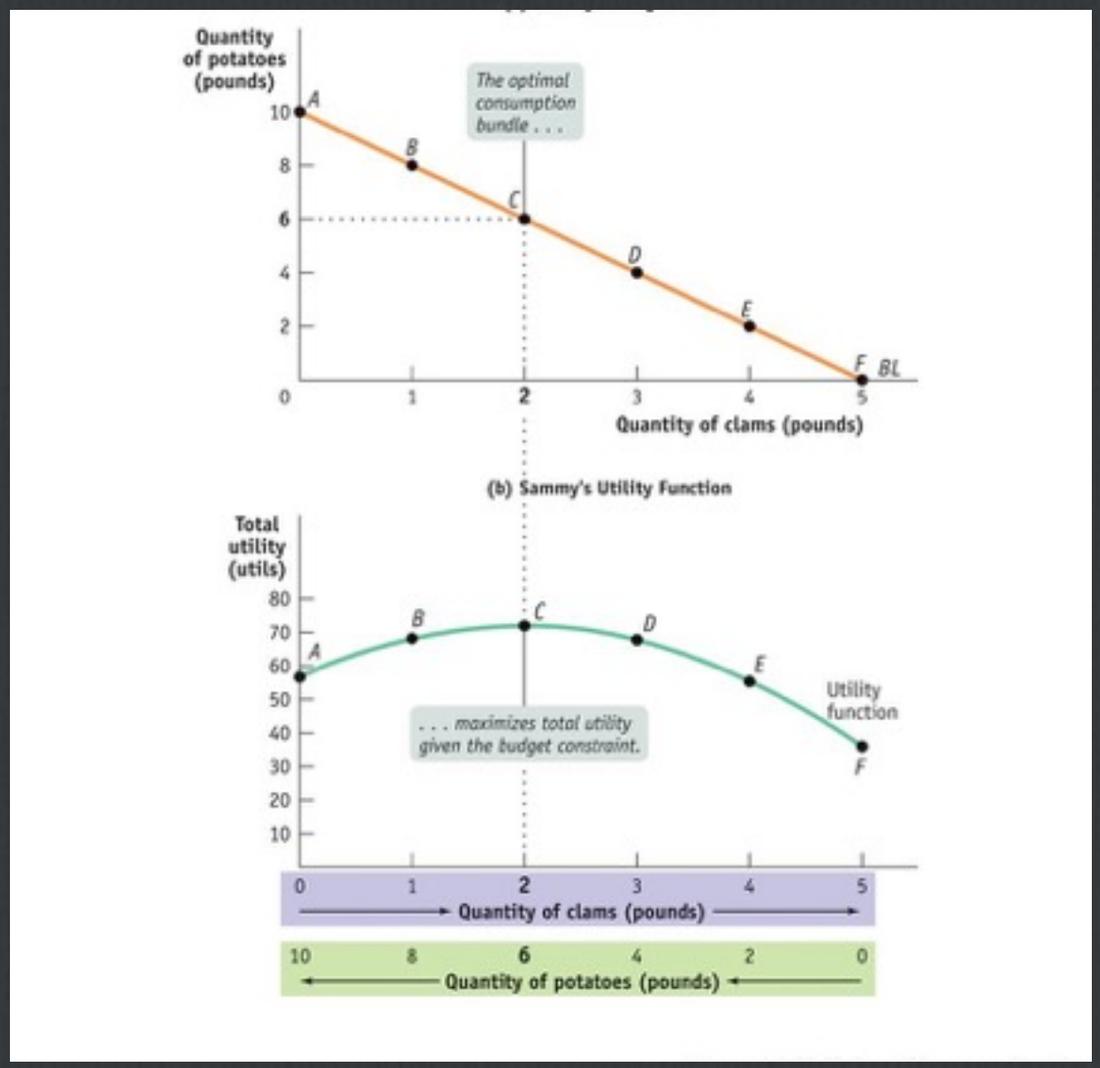
- ☐ You are thirsty
- ☐ 1 glass of water AMAZING
- ☐ + 1 glass of water, gooood
- ☐ + 1 glass of water, good
- ☐ + 1 glass of water, ok

Diminishing Marginal Utility



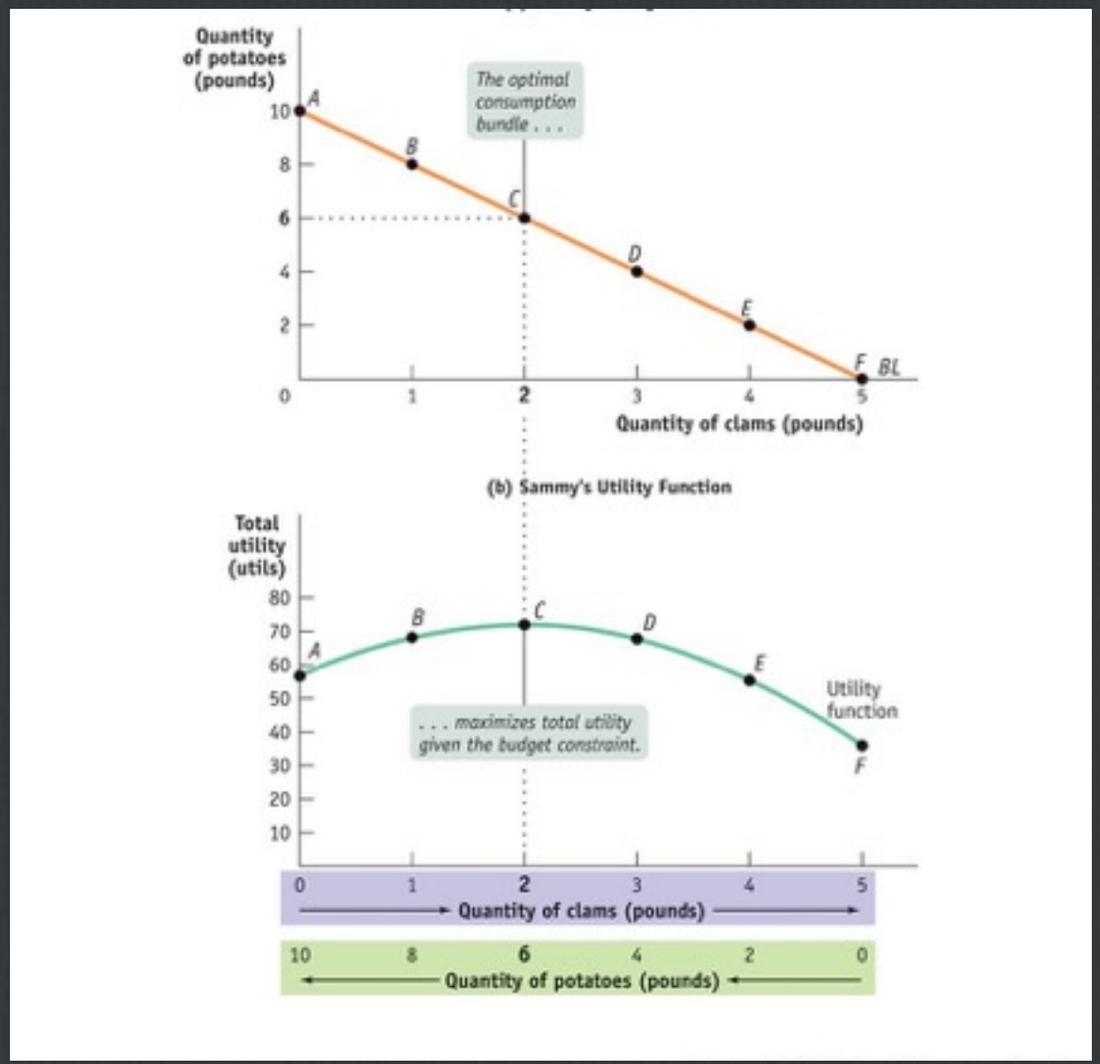
- ☐ + 1 glass of water, ENOUGH !
- ☐ + 1 glass of water, STOP !
- ☐ + 1 glass of water, Please STOP
- ☐ ....

Diminishing Marginal Utility



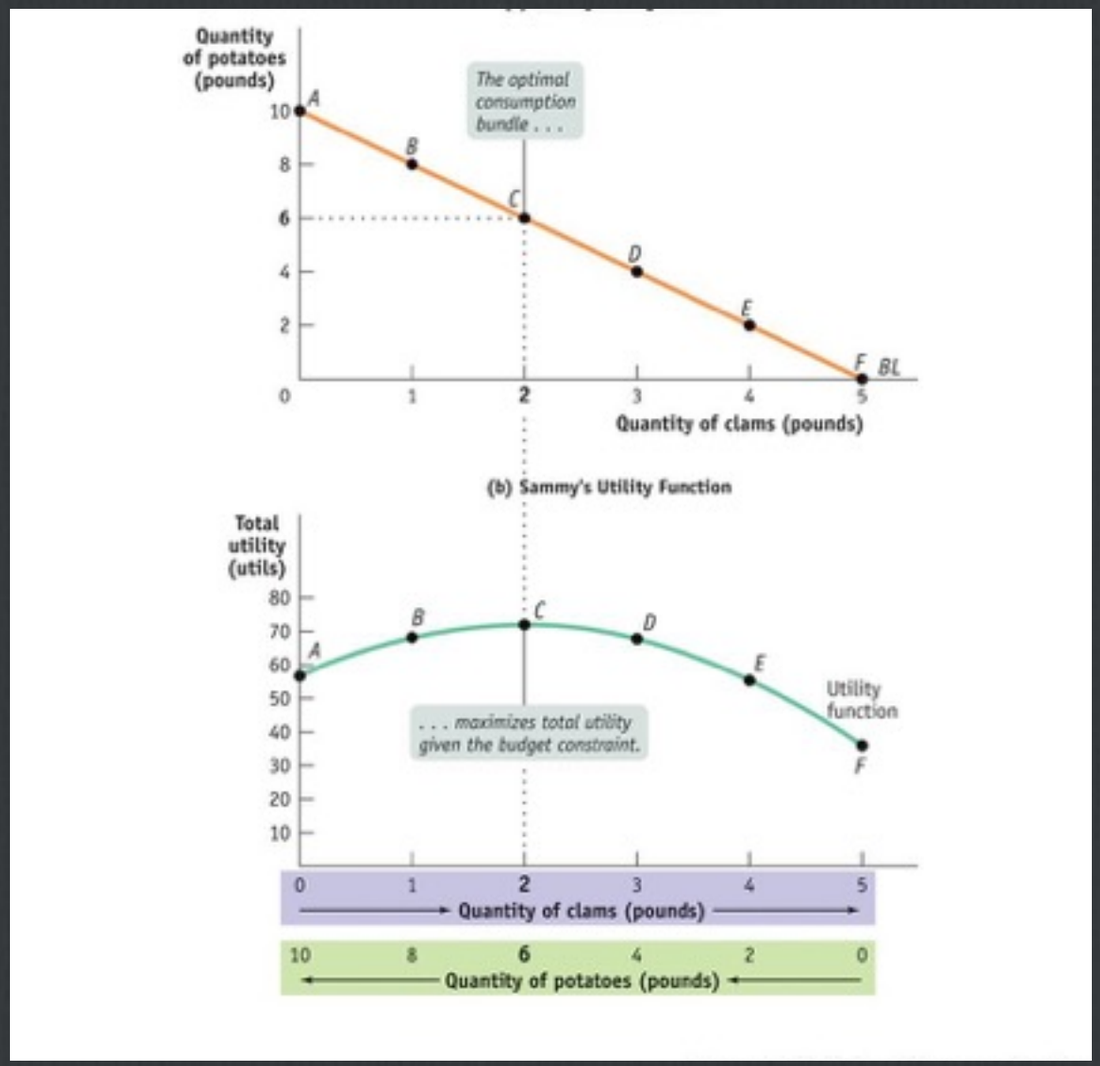
- Marginalists, ~1800 ADs
- Marginal utility decreases in every consumption
- Economics rule applies to many fields.

DMU: Work Hours !



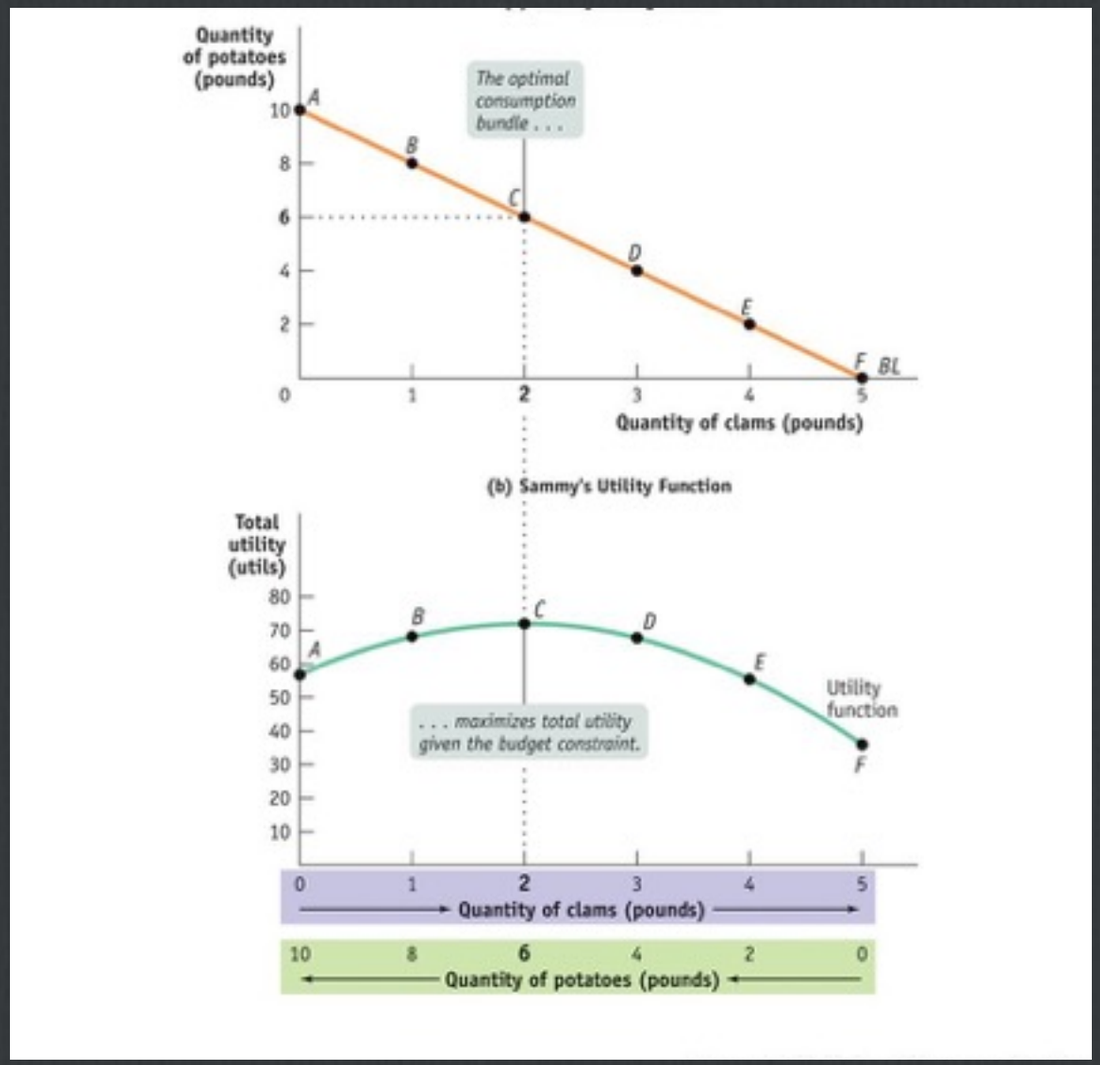
- ☐ First 2 hours, productivity is at tops
- ☐ Decreases in next 4 hours
- ☐ Lowers in next 2 hours
- ☐ Almost 0 at next 2 hours
- ☐ Starts producing bugs after 10 hours.
- ☐ One of the reasons many people thinks 4 hours is enough :)

DMU: Team Size !



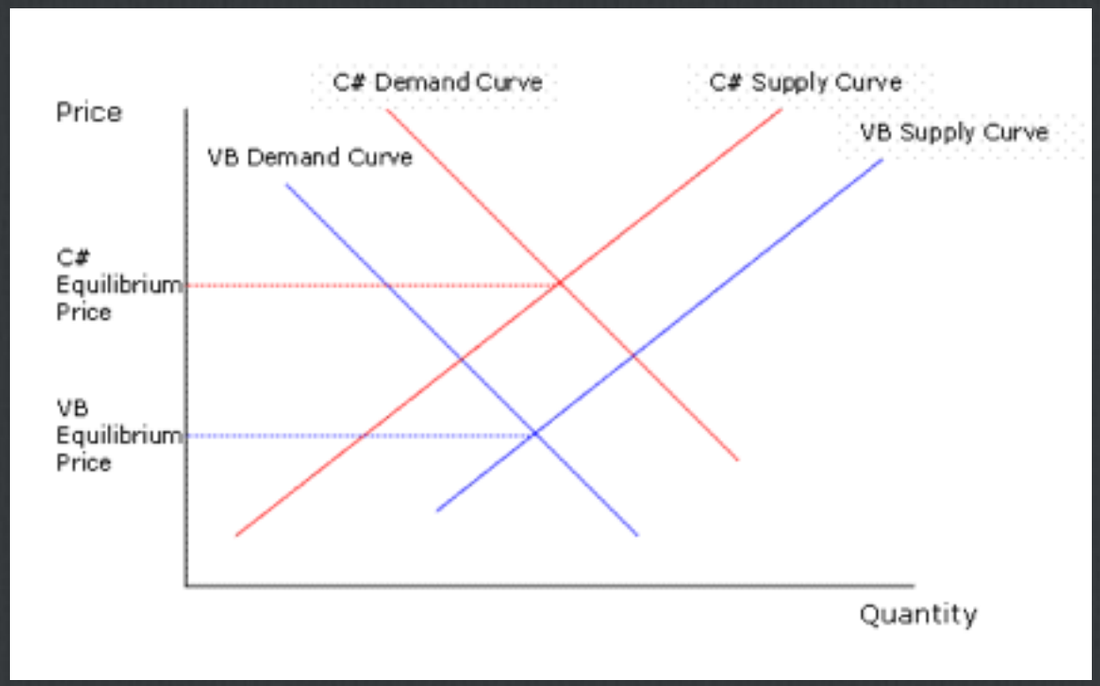
- ☐ Depending on project, adding 1 man adds lots of productivity
- ☐ add 1 more, more productivity
- ☐ add 1 more, more productivity but less than first 2
- ☐ add 1 more and you are saturated.
- ☐ add 1 more and you need managers. (or start splitting the team)

We used X Technology but now we'll switch



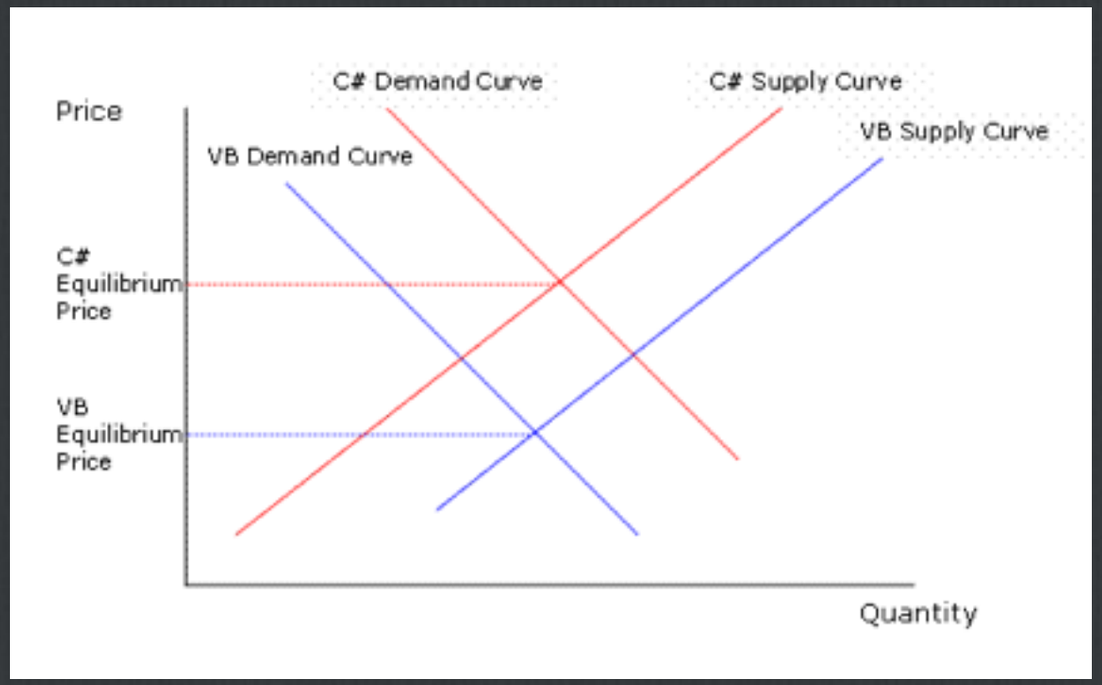
- We used X Technology but now we'll switch to Y Technology.
- We wrote 1000 lines was amazing, then we started…
- Our team was perfect but after adding feature X…
- Know where to stop !

Speaking of economics



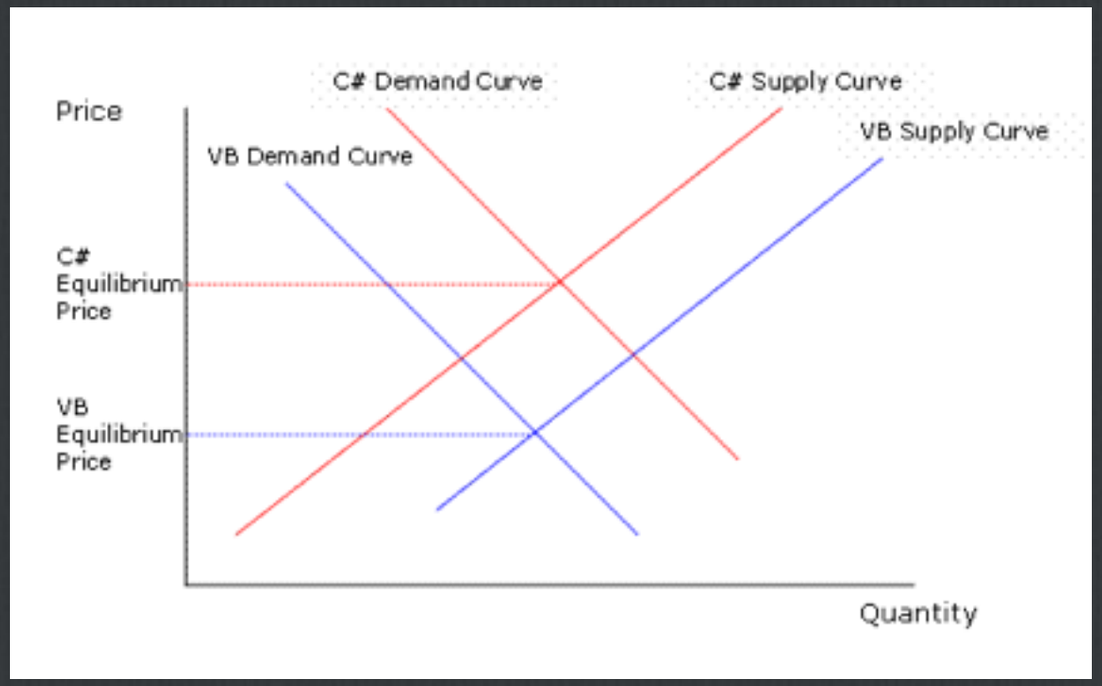
- ☐ Price theory
- ☐ Where supply and demand meets, happens the price.
- ☐ Choose your technology wisely.

Invest in yourself.



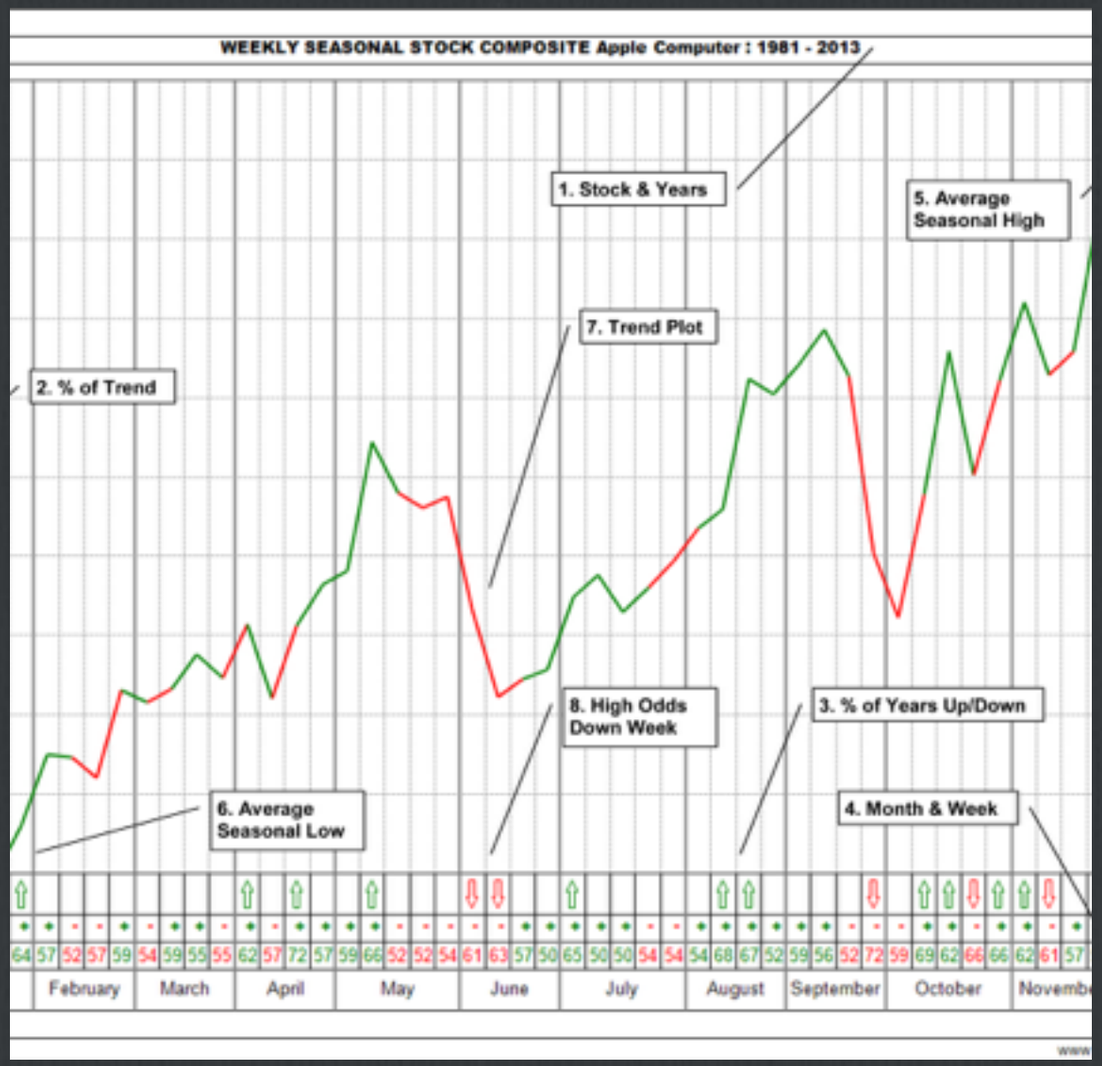
- ☐ you are actually selling your expertise and knowledge.
- ☐ If there is more demand to your X knowledge, your value rises.
- ☐ If there is less demand … well sorry.

Keep up with the pace ?



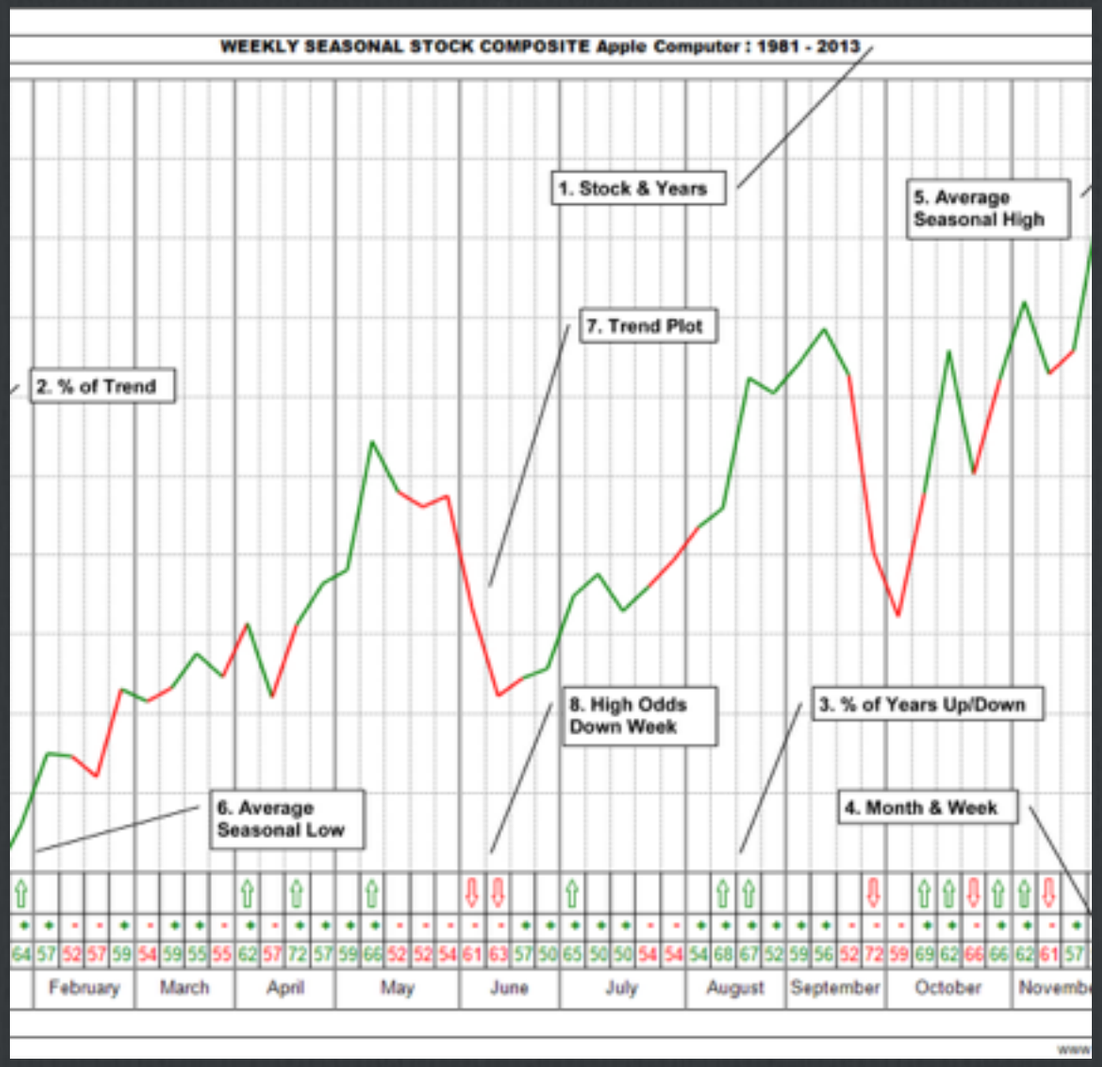
- ☐ Jump into every new tech because there is high demand ?
- ☐ Nodejs/Typescript ✔
- ☐ Bigdata ✔
- ☐ Bitcoin/Blockchain ✔
- ☐ C++17 ✔
- ☐ Fog Computing / IOT ✔
- ☐ Some shiny new tech ✔
- ☐ … ✔

Financial Investment



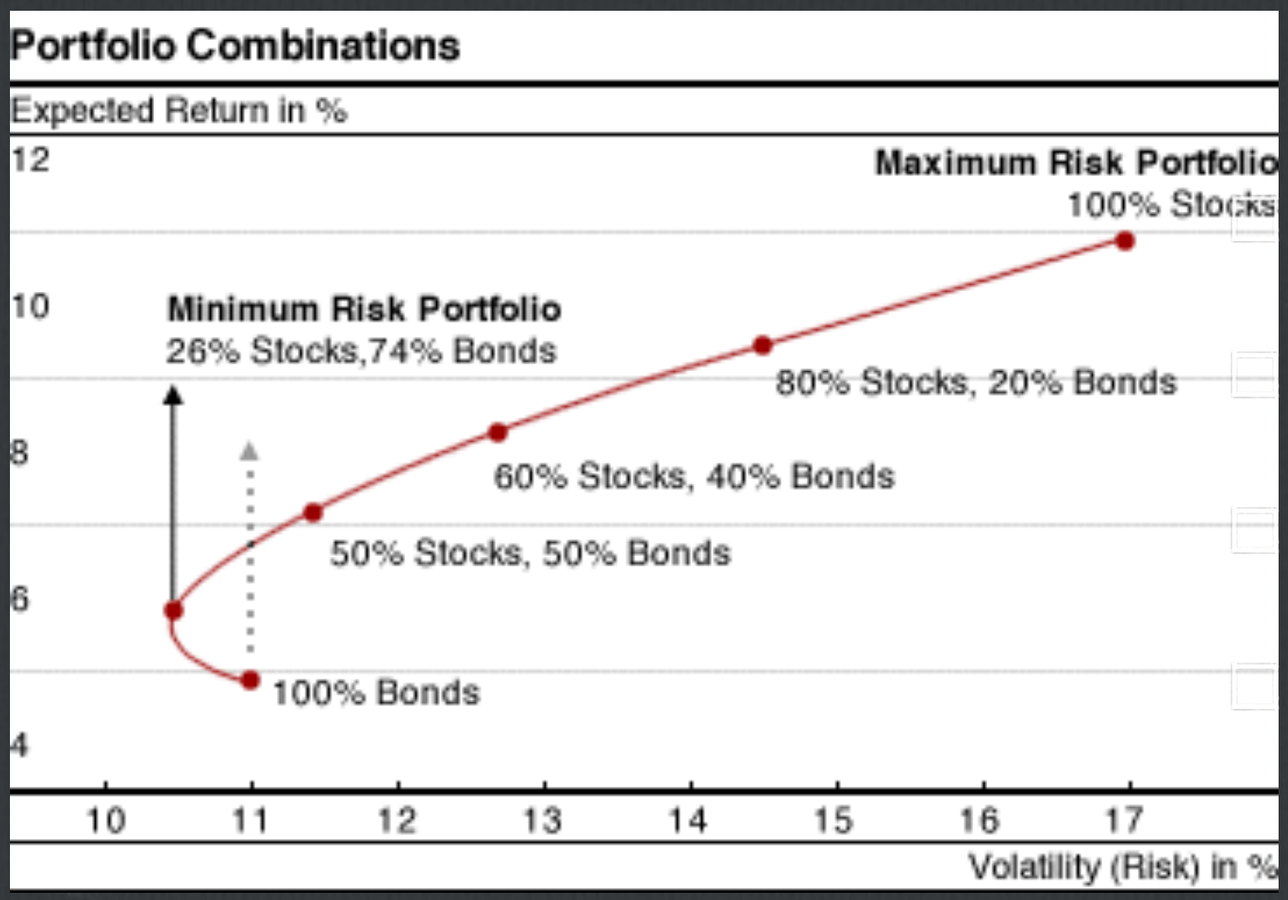
- ☐ A financial investment is an asset that you put money into with the hope that it will grow or appreciate into a larger sum of money.
- ☐ We have limited time instead of limited money.
- ☐ We don’t have bonds but technologies, programming languages.
- ☐ We put time on them…
- ☐ get some stock (as knowledge)
- ☐ then sell on the market when demand rises.

What if our shiny new tech is not that shiny anymore ?



- ☐ Demand falls, our value falls.
- ☐ We can’t learn everything
- ☐ Oh my…

Capital Asset Pricing Model



CAPM / Modern Portfolio Theory

Harry Markowitz / 1952

Won nobel with this.

Its about covariance

Covariance

In probability theory and statistics, covariance is a measure of how much two random variables change together.

some stocks tend to move together (because they are related or maybe not)

some tend to move with the market.

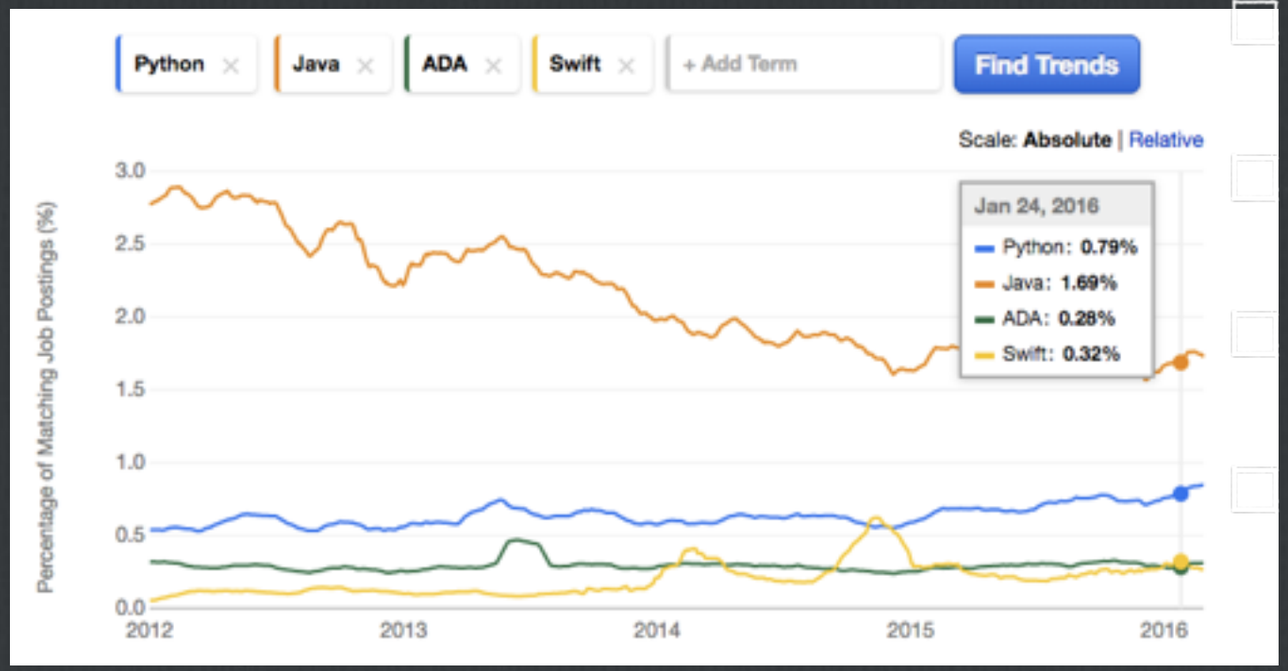
So choose your technology wisely.

Invest in yourself wisely.

Think about risk/return before you invest.

We have limited resources. Maximize your returns.

So choose your technology wisely.



Java/C++/Python low risk

Go lang, Swift trending

Pony lang… high risk

COBOL, Ada, Delphi low demand, but low supply too (so the price is high again)

What about Fun, Love & Hate…

- [ ] We all started this business because we love to code…

Behavioral economics

- ☐ Behavioral finance
- ☐ Markets are not that efficient
- ☐ Markets are sometimes irrational
- ☐ People are irrational

Behavioral economics

- ☐ Pagani Huayra
- ☐ 1,5 mil $
- ☐ Pay it and wait for 2 years, because they produce 40 per year
- ☐ Also you can’t drive it in the US up until recently
- ☐ 70% customers buy it, drive it near the factory and leave it at the factory - some buy a house near the factory and visit every year.

Behavioral economics

- Some people buy Apple stocks not because apple is sound, but because they hate Microsoft and love Iphone.
- So yes you can learn some Crystal, Pony lang… (or maybe brainfuck)

Brainfuck [edit]

```
++++++++++[>+++++++>++++++++++>+++>+
<<<<-]>++.>+.+++++++..+++.>++.<<+++++++++++++++.>.+++.------.--------.>+.>.
```

Also works as a Brainfork program.

What about efficiency ?

- We don't have “Leverage” as in finance
- If we can learn more in less time, then our investment options…

Learning Efficiency

- [ ] Learn something new every day
- [ ] Doesn’t matter what you learn
- [ ] Just make it become a habit
- [ ] Also Today I Learned list is cool.

Learning Efficiency

- ☐ Coursera…
- ☐ Lots of courses on history, music, economics, finance…
- ☐ Just make it become a habit
- ☐ also a Today I Learned list is cool.

Learning Efficiency

- ☐ History of rock’n’roll
- ☐ Rock’n’roll is rebellion, revolt ? Against everything…
- ☐ No. Black people were listening to radios, playing jazz, blues… White people were listening to radios, playing country…
- ☐ Radios wanted bigger audience so they could air more radio ads.
- ☐ They invented rock’n’roll, child of country and jazz and blues
- ☐ so they had more income.
- ☐ yes the music we listen today, most of it, is industrialized, and not pure.
- ☐ And this is what you learned today :)

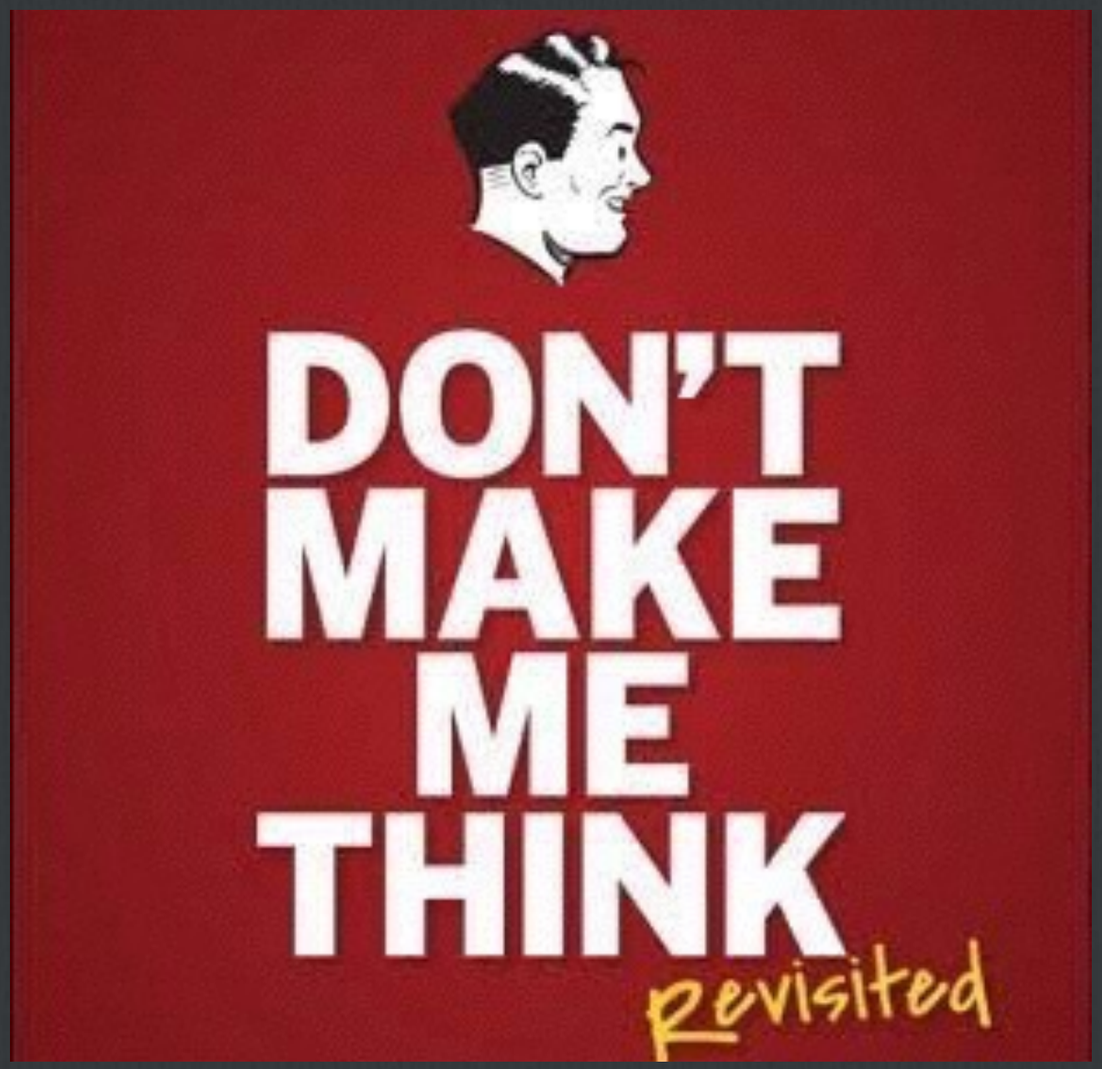
Learning Efficiency

- [ ] when learning new becomes a habit, you get used to it,
- [ ] so you don't really care much if a new version of your fav. framework is out and you have new docs to read.
- [ ] you start consuming faster.

Learning Efficiency

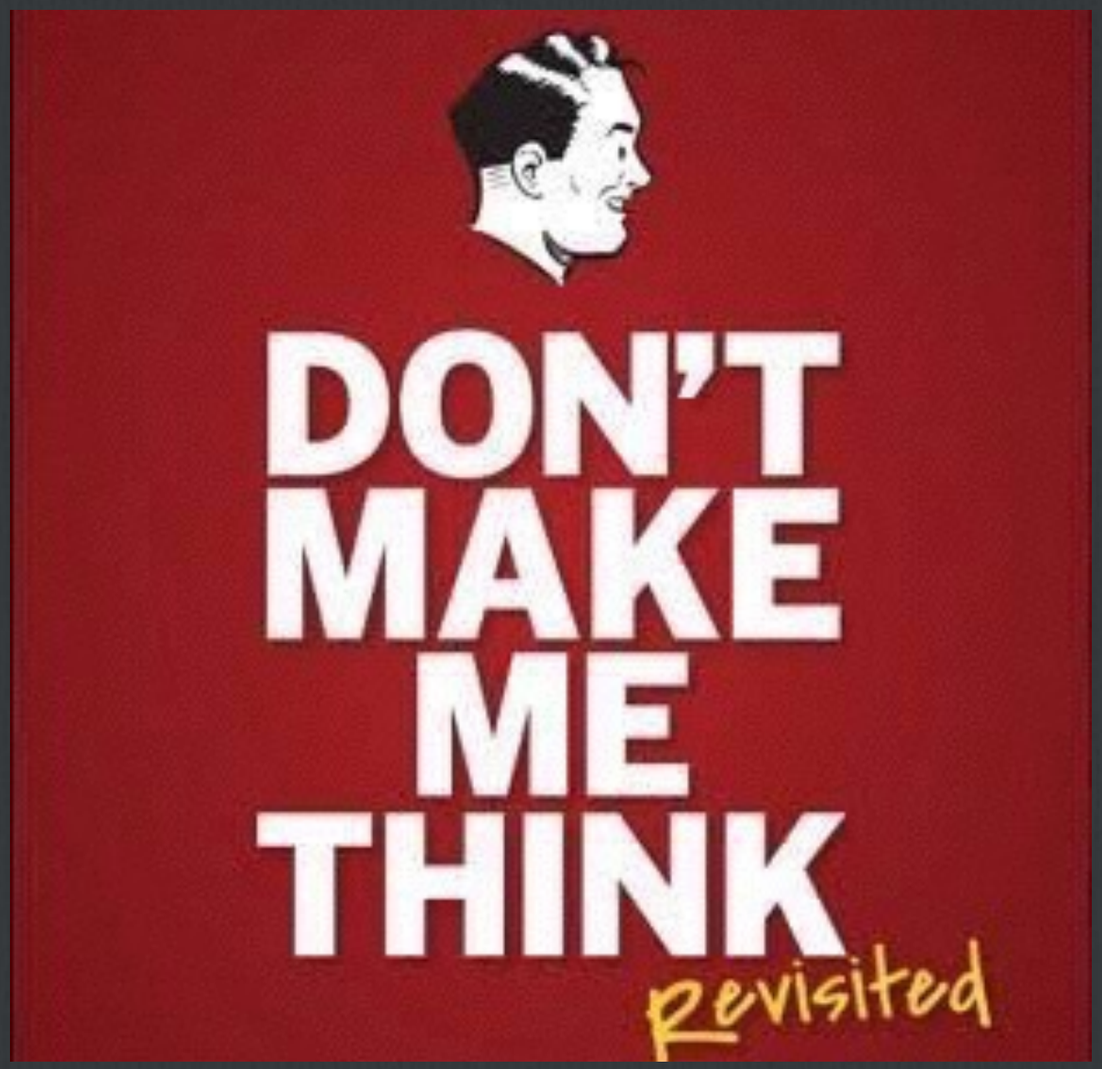
- [ ] Read some books
- [ ] Then read some more books

Don’t make me think



- Steve Krug
- Best book ever about programming (in my opinion)
- though its not about programming
- Don't Make Me Think is a book by Steve Krug about human–computer interaction and web usability.

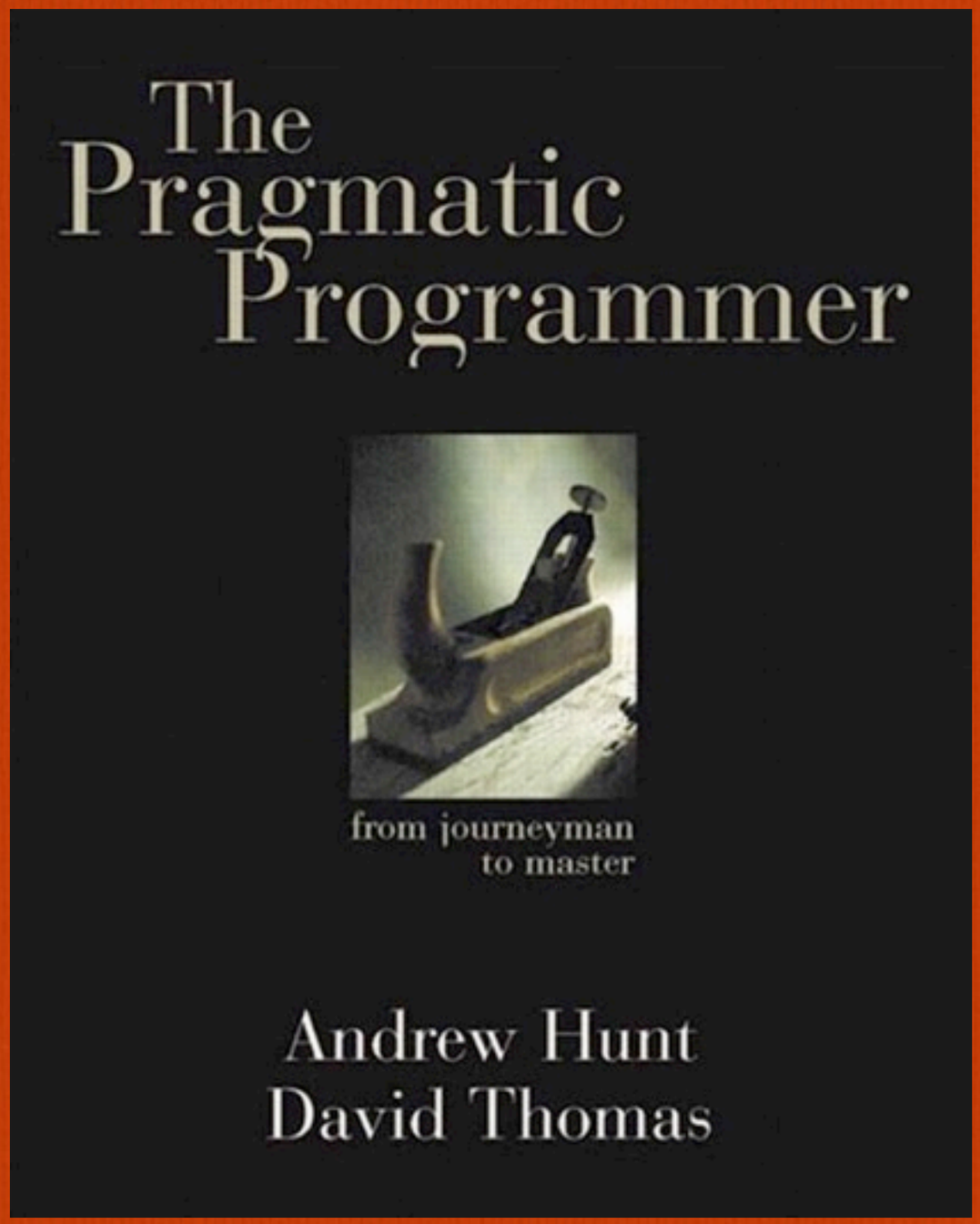
Programming is communication



- ☐ Does your code communicate ?
- ☐ Communicate with who ?
- ☐ Other programmers not the computer.

The Pragmatic Programmer from journey man to master

Andrew Hunt
David Thomas



Some Patterns from the book

- [ ] Broken Windows
- [ ] If you leave a broken window there, all system collapses.
- [ ] Fix it. Don’t look for someone to blame, just fix it.

Dead Programs Tell no lies.

- [ ] Crash early, crash often
- [ ] Don’t try to recover just let it crash.
- [ ] So you have one state, dead.

Good enough software

- [ ] You don’t have to be perfect.
- [ ] You usually can’t be perfect anyways.
- [ ] So ship something good enough, have feedback.

Tracer Bullets

- ☐ Don’t write the whole thing.
- ☐ Write some of it, fake some buttons, fake some actions.
- ☐ Eg. Have login, but no register, forgot password etc.
- ☐ Write the key features, but don’t try hard. Don’t try to be perfect.
- ☐ Ship it, show it. If everything ok go for it.

Read More

- [ ] Also recommended reading,
  - [ ] Refactoring, Martin Fowler
  - [ ] Clean Code, Robert C. Martin
  - [ ] Code Complete, Steven C. McConnell

Learn some new concepts

Clauses with empty bodies are called **facts**

```
cat(tom).
```

which is equivalent to the rule:

```
cat(tom) :- true.
```

The built-in predicate `true/0` is always tr

Given the above fact, one can ask:

is tom a cat?

```
?- cat(tom).
Yes
```

what things are cats?

```
?- cat(X).
X = tom
```

- [ ] Prolog
- [ ] Logic Programming

Learn some new concepts

```
382 parse_uri_path(<< C, Rest/bits >>, State, Method, SoFar) ->
383         case C of
384                 $\r -> error_terminate(400, State, {connection_error, protocol_error,
385                         ''}); %% @todo
386                 $\s -> parse_version(Rest, State, Method, SoFar, <<>>);
387                 $? -> parse_uri_query(Rest, State, Method, SoFar, <<>>);
388                 $# -> skip_uri_fragment(Rest, State, Method, SoFar, <<>>);
389                 _ -> parse_uri_path(Rest, State, Method, << SoFar/binary, C >>)
390         end.
391
392 parse_uri_query(<< C, Rest/bits >>, State, M, P, SoFar) ->
393         case C of
394                 $\r -> error_terminate(400, State, {connection_error, protocol_error,
395                         ''}); %% @todo
396                 $\s -> parse_version(Rest, State, M, P, SoFar);
397                 $# -> skip_uri_fragment(Rest, State, M, P, SoFar);
398                 _ -> parse_uri_query(Rest, State, M, P, << SoFar/binary, C >>)
399         end.
```

- ☐ Functional Programming
- ☐ ML, Lisp, Haskell, Erlang
- ☐ Choose one.

Learn some new concepts

```
procedure Push (S: in out Stack; E: in Element)
 with Pre => Not_Full (S),
      Post => (S.Length = S'Old.Length + 1) and
              (S.Data (S.Length) = E) and
              (for all J in 1 .. S'Old.Length =>
                 S.Data (J) = S'Old.Data (J));
```

- ☐ Design by contract
- ☐ Eiffel, Ada, D

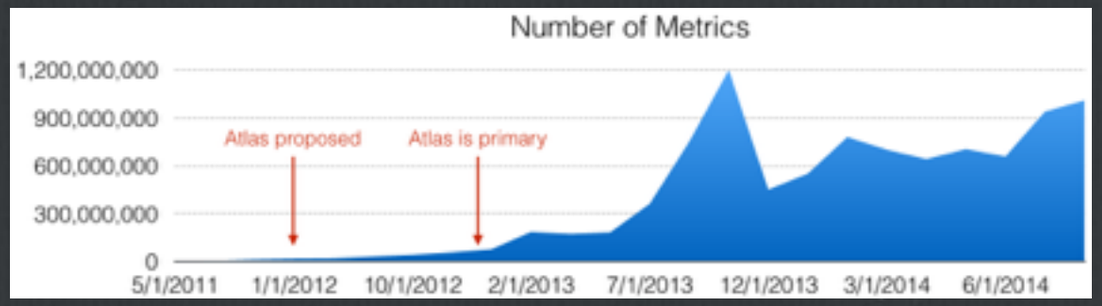
More Concepts

- [ ] DevOps
- [ ] What happened to sysadmins ?

More Concepts

- [ ] Immutable Infrastructures
  - [ ] docker, rkt, containers
  - [ ] deploy code or servers
- [ ] Serverless deployment (amazon lambda…)
- [ ] Cloud, auto scale…

More Concepts



- Monitoring
  - Netflix
  - Collecting 1billion metrics

More Concepts

- ☐ Continous Delivery
  - ☐ Continous Integration
  - ☐ Continous Deployment
- ☐ 2 week sprints vs. Try deploying every 3 minutes

More Concepts

- [ ] Microservices
- [ ] Small services are 100 lines of code, do you really need to test them ?
- [ ] Testing vs. Monitoring

More Concepts

- [ ] Message passing
- [ ] Pub/Sub, Event Stream, Fanout, Pipeline
- [ ] zmq documentation
- [ ] http://zguide.zeromq.org/page:all#sockets-and-patterns

Learn Concepts

- ☐ Not technologies, technologies evolve, change, die
- ☐ Concepts die harder :)
  - ☐ it was corba
  - ☐ now its microservices
  - ☐ its all about message passing

Learn Concepts

- [ ] Any new slide from now on will have zero or less marginal utility
- [ ] Diminishing Marginal Utility
  - [ ] :)

“So what is a pragmatic programmer ??”

Pragmatic Programmer

- ☐ They,
- ☐ care about their work, and accepted responsibility for it
- ☐ are always learning, they enjoy change
- ☐ value the depth and breadth of their experience, and are constantly looking for ways of applying it to the issues at hand
- ☐ are highly aware of what they were doing
- ☐ treat analysis, design, implementation, testing, and support as different facets of an overall process, rather than as discrete activities.

oh, and one last slide…

DON'T

PANIC