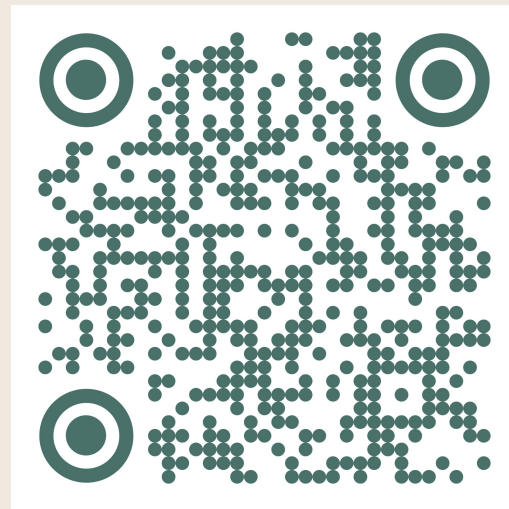


Det våras för $f(x)$

Konsten att designa för förändring



mint.se/det-varas-for-funk



Johan Burell



Systemarkitekt
Mint

Om mig

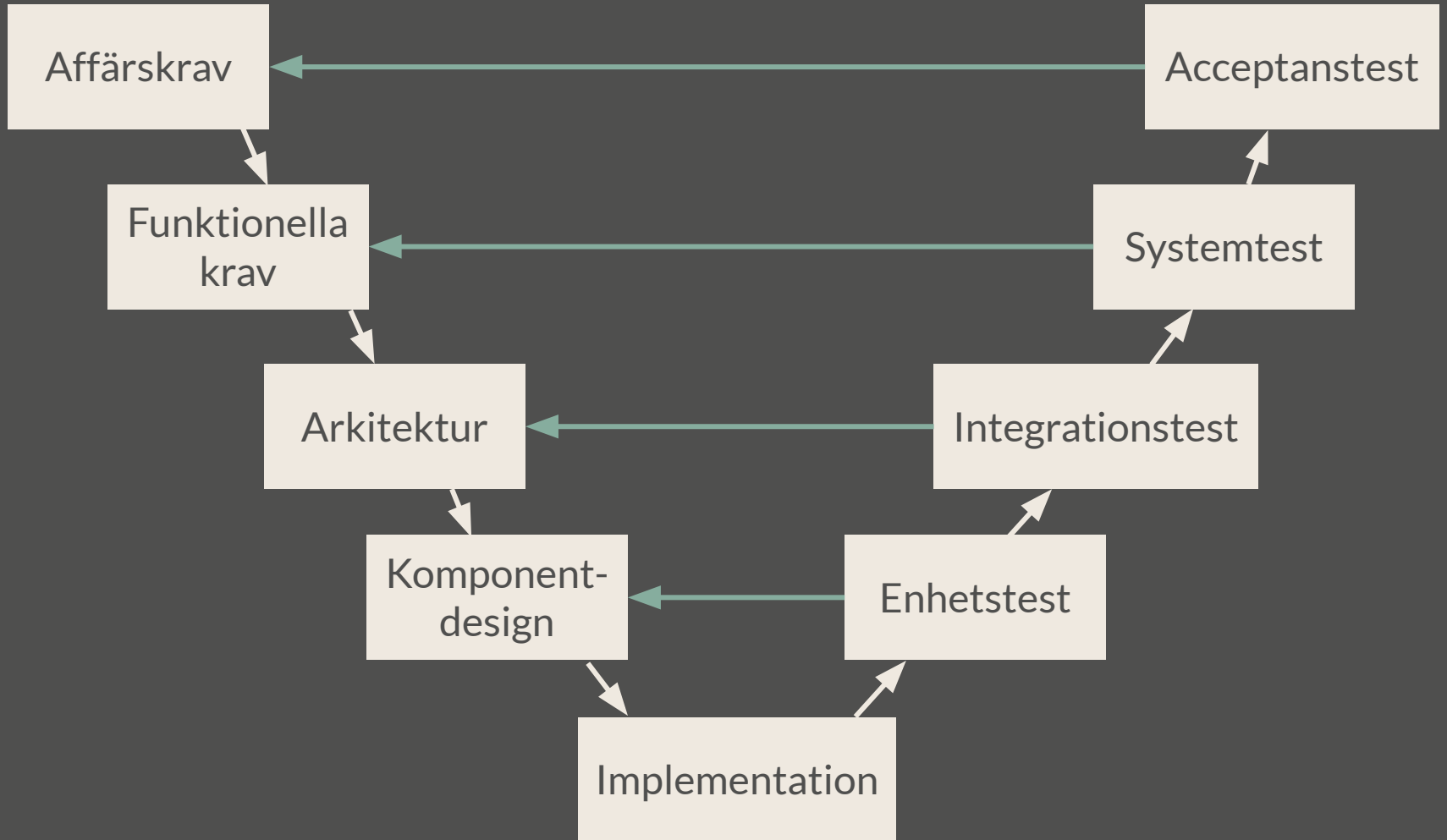
- Arbetat 21 år som utvecklare/arkitekt
- Spelutveckling, distribuerade system, frontend, backend, operations...
- Funktionell programmering: ca 6 år
- .NET (F#/C#), Java, Kotlin, Rust, C++...
- Brinner för Craftsmanship-kultur, som
 - Agilt arbete
 - Testdriven utveckling
 - Domändriven design
 - ...

Den mänskliga faktorn

**Människans korttidsminne beräknas att klara
av 7 ± 2 variabler samtidigt.**

Så hur kan det existera “10x utvecklare”???

Hygienfaktorer



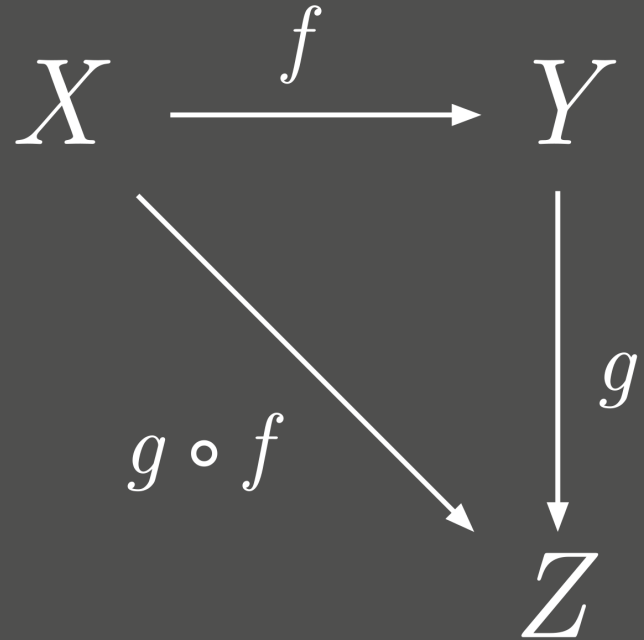
*“Testing can be used to show the presence of bugs,
but never to show their absence.”*

- Edsger W. Dijkstra



Formell verifikation och
kategoriteori levererar
garantier bortom test

...och med snabbare
feedback



Funktionell programmering är inte ett verktyg, det är ett sätt att förhålla sig till problem.

Abstraktioner genom komplexa kopplingar...

**...föder sammanvävda komponenter och
“magiska beteenden”.**

Simple made easy (Hickey, via InfoQ)



Download MP3

01:01:26

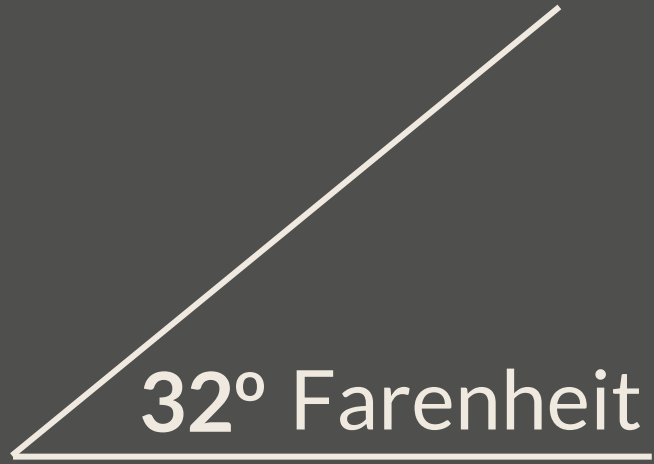
Summary

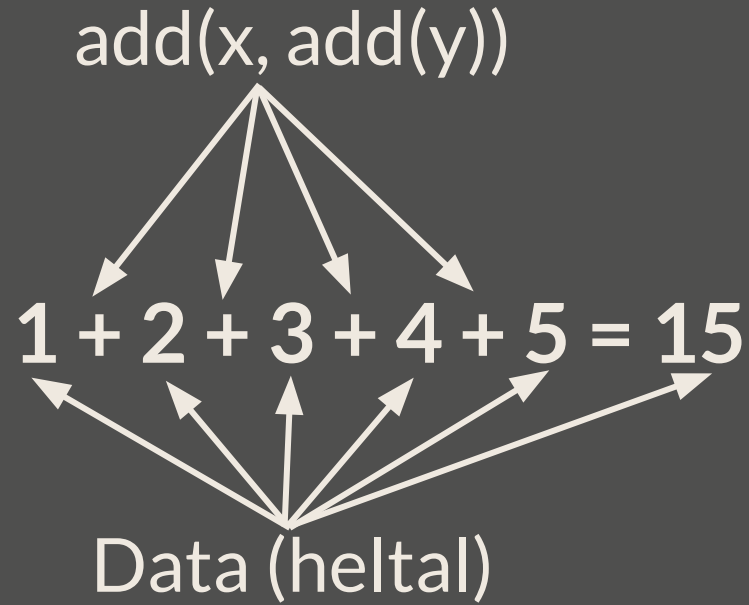
Rich Hickey emphasizes simplicity's virtues over easiness, showing that while many choose easiness they may end up with complexity, and the better way is to choose easiness along the simplicity path.

Simple

- One fold/braid
- One role
- One task
- One concept
- One dimension
- But not
 - One instance
 - One operation
- About lack of interleaving, not cardinality
- *Objective*

Typer





Prestanda

FP är normalt sett långsammare och mer minneskrävande än OOP/imperativ kod.

mandelbrot

source	secs	mem	gz	cpu secs
<u>F# .NET #6</u>	3.83	140,068	933	14.86
<u>F# .NET</u>	4.08	140,580	877	15.88
<u>F# .NET #5</u>	4.09	139,916	897	15.90
<u>Java #4</u>	4.39	101,236	660	16.12
<u>Java #2</u>	4.11	83,924	796	16.21
<u>Java #6</u>	4.27	84,532	802	16.86
<u>Java</u>	25.66	40,860	665	25.73
<u>Java #3</u>	7.37	88,056	903	29.24

×	source	secs	<u>mem</u>	<u>gz</u>	<u>cpu secs</u>	cpu load			
1.0	C gcc #9	0.78	11,284	1463	0.78	14%	85%	0%	0%
1.0	Chapel #6	0.79	10,880	1104	0.80	100%	0%	1%	0%
1.0	C gcc #2	0.81	11,392	2268	3.23	100%	100%	98%	100%
1.1	C gcc #7	0.84	11,392	2011	3.26	96%	96%	96%	100%
1.2	Rust #5	0.90	11,060	1961	3.08	85%	84%	89%	84%
1.3	C++ g++ #5	1.04	10,940	2344	3.66	88%	87%	88%	88%
1.4	Julia #8	1.12	191,000	1082	1.38	7%	8%	99%	8%
1.5	F# .NET #3	1.16	137,720	1350	4.13	92%	86%	86%	92%
1.5	Java #6	1.16	46,068	2543	3.65	97%	70%	73%	72%
1.6	Java #5	1.23	46,180	2473	3.79	70%	91%	76%	72%
1.6	Go #2	1.26	11,136	1404	4.26	83%	84%	89%	84%
1.6	F# .NET #4	1.29	201,344	1342	4.54	85%	89%	85%	92%
1.7	C gcc #5	1.30	11,392	1281	1.30	0%	100%	0%	0%

*“Vi måste sluta säga till folk att FP
är bra för data engineering...”*

Bemanning

**Anställ inte bara FP-utvecklare,
våga konvertera OOP-utvecklare**

4,5 år

20+ utvecklare

0 år erfarenhet av F#

...100% framgångsrikt

Att lära sig en ny domän tar tid...
Att lära sig en ny produkt tar tid...
Att lära sig gruppdynamik tar tid...
Att förstå kunden tar tid...

Pragmatism

Börja smått
Ha ett mål
Ha en strategi
Gör kompilatorn till din vän

*“Better a diamond with a flaw
than a pebble without one.”*

- Kinesiskt ordspråk

Partners

FUNCSOFT



<https://functionalsoftware.se/>

Ada
Beat



<https://adabeat.com/insight/why-functional-programming-for-ceos/>

Why F# is the best enterprise language

20 Dec 2018

This post is part of the [2018 F# Advent Calendar](#). Check out all the other great posts there! And special thanks to Sergey Tihon for organizing this.

“Why F# is the best enterprise language” is not meant to be a clickbait title – it is my sincere opinion, and in this post I will attempt to justify it. If you stick around to the end, I hope that you will agree, or at least be a little bit persuaded. Read on!



<https://fsharpforfunandprofit.com/posts/fsharp-is-the-best-enterprise-language>

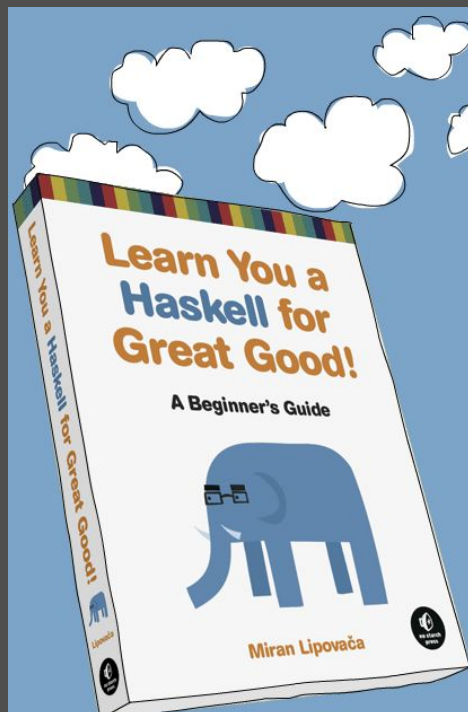
/

Domain Modeling Made Functional

Tackle Software Complexity with
Domain-Driven Design and F#



<https://www.pragprog.com/titles/swdddf/domain-modeling-made-functional/>



Hey yo! This is **Learn You a Haskell**, the funkiest way to learn Haskell, which is the best functional programming language around. You may have heard of it. This guide is meant for people who have programmed already, but have yet to try functional programming.


The whole thing is completely free to read online, but it's also available in print and I encourage you to buy as many copies as you can afford!

To contact me, shoot me an email to: [bonus at learnyouahaskell dot com](mailto:bonus@learnyouahaskell.com)! You can also find me idling on [#haskell](https://twitter.com/haskell) where I go by the name **BONUS**.

Got questions? READ THE FAQ

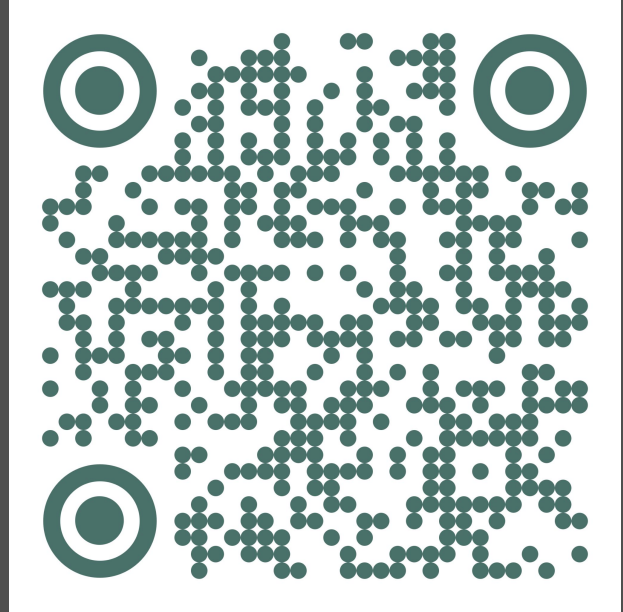
Buy it!
You know you want to!

Read it online!
For free!



<http://learnyouahaskell.com/>

Tack!



mint.se/det-varas-for-funk

Objekt

AnvändarObj

AuthToken

Profil {Namn, Bild, ...}

*AnvändarObj(
 LoginHanterare,
 ProfilHanterare,
 MeddelandeHanterare,
) -> AnvändarObj*

loggaIn(ID,Lösen) -> Token

profil() -> Profil

posta(Msg) -> _

Funktioner

LoginModul

Login {ValidID, ValidLösen}

valideraLogin(ID,Lösen) -> Result<Login, Err>

loggaIn(Hndl,LoginData) -> Result<Token, Err>

ProfilModul

Profil { Namn, Bild }

profil(Hndl,Token) -> Result<Profil, Err>

MeddelandeModul

posta(Hndl,Token,Msg) -> Result<_, Err>

Objekt

```
var user = AnvändarObj(  
  LoginHanterare,  
  ProfilHanterare,  
  MeddelandeHanterare,  
);  
  
user.loggaIn(ID,Lösen);  
  
var profile = user.profil();  
  
user.posta("hejsan");
```

Funktioner

```
let loggaInHanterare = loggaIn(Hndl)  
  
let tokenOrError =  
  valideraLoginData(ID,Lösen)  
  >=> loggaInHanterare  
  
let profilHanterare = profil(Hndl)  
  
let profileOrError =  
  tokenOrError  
  >=> profilHanterare  
  
let postHanterare =  
  tokenOrError  
  >=> post(Hndl)  
  
let result = postHanterare("hejsan")
```