

Discover IT

Leren programmeren?

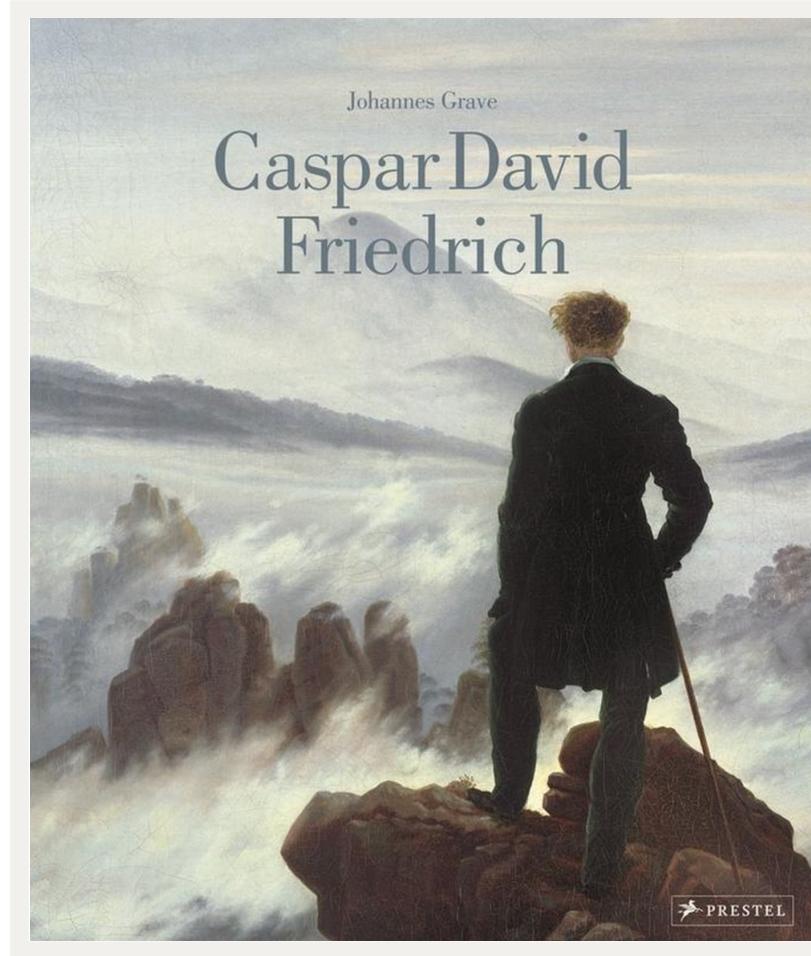
**Kristof Michiels**

# Het zicht vanop de top...



... is prachtig?

# Het zicht vanop de top...



... is prachtig?

# Leren programmeren? Je diploma behalen?



Het beklimmen van een berg?

# There will be sherpa's



Uw nederige docent ;-)

# Je bereikt een aantal basecamps



Is iedereen nog mee?

# Niets is 100% wat het lijkt



Rommel onderweg...

# Niets is 100% wat het lijkt



File onderweg...

# Hoe het ook zij: leren programmeren is moeilijk

- Jullie kennen mijn aanpak voor *Web Technology*
- Hoe leert iemand programmeren?
- Welke vaardigheden en attitudes zijn daarvoor nodig?
- Hoe oefen je?

# De aanpak bij *Web Technology*



10 / 41 "Babies thrown in the deep end at Dingley's Kids Aquatic Survival School in bid to prevent drowning"

# De aanpak bij *Web Technology*

- Begeleidende context maar er snel ingevlogen
- Daar is over nagedacht: <https://www.w3.org/Style/CSS/all-properties.en.html> (> 500 eigenschap-namen)
- Oefeningen:
  - Strak geleid, gaandeweg krijg je meer vrijheid
  - We beginnen langzaam te diversifiëren
  - Met (vrijblijvende) uitdagingen (<https://cssbattle.dev/> anyone? ;-))

# Verschillen in aanpak

- Jullie krijgen te maken met verschillende technologieën, vakken, docenten, aanpakken, ...
- Daar is over nagedacht: door beleid, school, opleiding, docent, student, ...
- Toch kan het zijn dat de ene aanpak of route je beter ligt dan de andere

# Hoe leert iemand programmeren?

# Hoe leert iemand programmeren?

- De top-down aanpak

- Snel iets realiseren door echt te programmeren
- bvb een (gedetailleerde) tutorial volgen, stap voor stap
- Kan aanstekelijk, bevestigend werken, aanzetten tot verder leren en ontdekken
- Je hoeft niet elk aspect van een programmeertaal kennen om productief te zijn
- Veel concepten tegelijkertijd
- Kennis van de fundamentele concepten kan erbij inschieten

# Hoe leert iemand programmeren?

- De bottom-up aanpak

- Eerst focussen op de fundamentele concepten van programmeren
- Starten op nul en - gestructureerd - concept per concept leren kennen
- Grondige basis verwerven die nuttig is bij elke vorm van programmeren, in welke taal of omgeving dat ook gebeurt
- Elk individueel - geïsoleerd - concept is gemakkelijker aan te leren
- Traag, soms vervelend: duurt lang voor je iets wezenlijks maakt

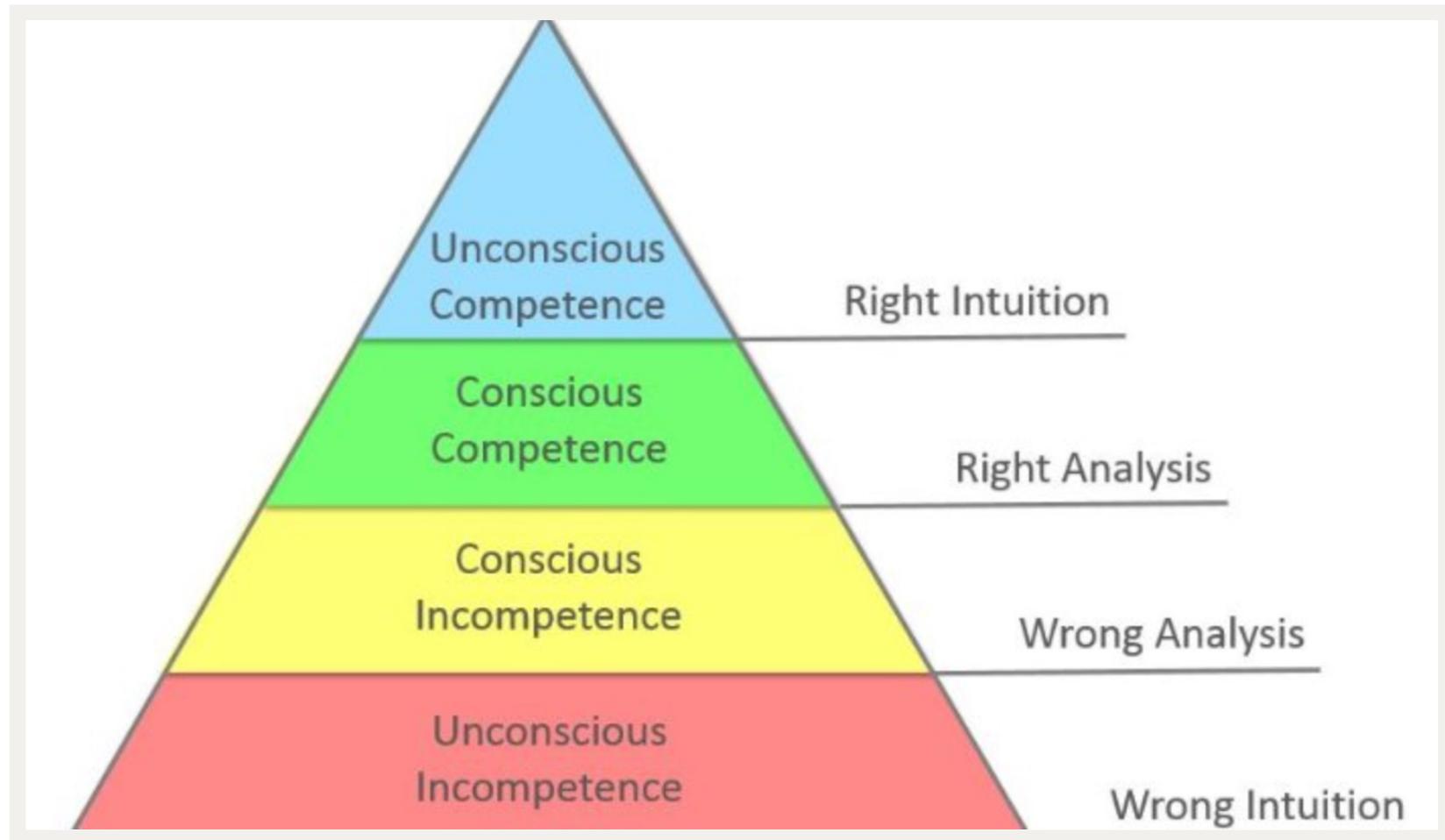
# Hoe leert iemand programmeren?

- Om een goede programmeur te worden is een mix van beide aanpakken nodig
- Software maken zonder volledig begrip is belangrijk
- Kennis van de fundamenteen is even belangrijk
- Het leerproces eindigt nooit
- Wissel af tussen beide aanpakken
- Software ontwikkeling is niet eenvoudig: niemand kent alles en er is altijd iets te leren
- Neem je tijd en geniet van de weg naar boven

# Moeilijk: hoe meet je leervooruitgang?

- Wanneer ken je een programmeertaal?
  - Als je de syntax kent?
  - Als je met de vingers in de neus populaire bestaande *libraries* kunt gebruiken in je code?
  - Of wanneer je alle details kent van het hele ecosysteem rond een bepaalde taal?
- Categorieën: beginner, gevorderd, expert?
- Nuttiger: 4 stadia van competentie: [https://en.wikipedia.org/wiki/Four\\_stages\\_of\\_competence](https://en.wikipedia.org/wiki/Four_stages_of_competence)

## 4 stadia van competentie



[https://en.wikipedia.org/wiki/Four\\_stages\\_of\\_competence](https://en.wikipedia.org/wiki/Four_stages_of_competence)

## 4 stadia van competentie

- Onbewust onbekwaam: je mist kennis of vaardigheden op een bepaald gebied, maar bent je er ook niet van bewust dat je deze nodig zou hebben
- Bewust onbekwaam: je bent je er pijnlijk van bewust wat je allemaal niet kan. Iedereen om je heen competenter te zijn. Dit ondermijnt je zelfvertrouwen en kan er zelfs voor zorgen dat je helemaal opgeeft
- Bewust bekwaam: je hebt de vaardigheden eigengemaakt, maar moet je nog steeds bewust concentreren om ze goed uit te voeren. In dit stadium is het handig om jezelf zoveel mogelijk in situaties te brengen waarin je de vaardigheden kan trainen
- Onbewust bekwaam: je gebruikt je vaardigheden moeiteloos en kunt je niet meer voorstellen dat je het ooit niet hebt gekund. In dit stadium kun je veel leren door de vaardigheden ook aan andere mensen over te dragen

## 4 stadia van competentie: beter inzicht in leerproces

- Inzicht in de 4 stadia helpt om je emoties te begrijpen tijdens je eigen leerproces
- Een coach kan een lerende door de verschillende stadia helpen:
  - onderzoeken van zwaktes die versterkt moeten worden
  - opbouwen van zelfvertrouwen
  - oefenen om beter te worden
  - onderhouden van de vaardigheid

# Welke vaardigheden en attitudes zijn nodig?

# Vaardigheden en attitudes? Heel wat! Hard en zacht

Werden reeds genoemd door Marc:

- Relateren en structureren
- Kritisch verwerken
- Concretiseren
- Analyseren

# Tao of programming

## *The Tao of Programming*

Translated by Geoffrey James

*Note: I copied this from <http://misspiggy.gsfc.nasa.gov/tao.html> and stripped out all of the IMHO extraneous formatting.  
--Alex*

### BOOK 1

#### *The Silent Void*

Thus spake the Master Programmer:

**"When you have learned to snatch the error code from the trap frame, it will be time for you to leave."**

##### 1.1

Something mysterious is formed, born in the silent void. waiting alone and unmoving, it is at once still and yet in constant motion. It is the source of all programs. I do not know its name, so I will call it the Tao of Programming.

If the Tao is great, then the operating system is great.  
If the operating system is great, then the compiler is great.  
If the compiler is great, then the application is great.  
The user is pleased, and there is harmony in the world.

The Tao of Programming flows far away and returns on the wind of morning.

##### 1.2

The Tao gave birth to machine language. Machine language gave birth to the assembler.

The assembler gave birth to the compiler. Now there are ten thousand languages.

Each language has its purpose, however humble. Each language expresses the Yin and Yang of software. Each language has its place within the Tao.

But do not program in COBOL if you can avoid it.

<https://www.mit.edu/~xela/tao.html>

# Vaardigheden en attitudes? Heel wat! Hard en zacht

- Motivatie
- Nieuwsgierigheid
- Zelfvertrouwen
- Geduld en doorzettingsvermogen
- Computationeel denken

Een programmeertaal leren lijkt nog het gemakkelijkste... Hoe stop je die zaken in een opleiding?

# Computationeel denken

Ik licht er hier één van de allerbelangrijkste uit

- Leren denken (als een computer)
- Geen resultaat van specifieke kennisopbouw
- 4 kernonderdelen:
  - ontleding: van complexere problemen in kleinere eenvoudigere problemen
  - patroonherkenning: verbanden leggen tussen gelijkaardige problemen en ervaringen
  - abstractie: belangrijke informatie identificeren en niet-gerelateerde of irrelevante details negeren
  - algoritmen: met eenvoudige stappen problemen oplossen

# Computationeel denken

Gelukkig kunnen we dit leren!

Daarom: oefening baart kunst!

# Hoe oefen je computationeel denken?

- ontleding: aanbieden van probleem-scenario's. De leraar deelt het complexe, uit meerdere stappen bestaande probleem en faciliteert gesprekken die helpen om het op te lossen
- patroonherkenning: studenten confronteren met verschillende patronen en ervaringen die de gemeenschappelijkheden laten zien
- abstractie: studenten activeren om zelfstandig op zoek te gaan naar een oplossing, hierbij een bredere set van bronnen aanbiedend
- algoritmen: aanmoedigen en helpen bij het aanpakken van een probleem terwijl hierbij een reeks te nemen stappen wordt afgesproken

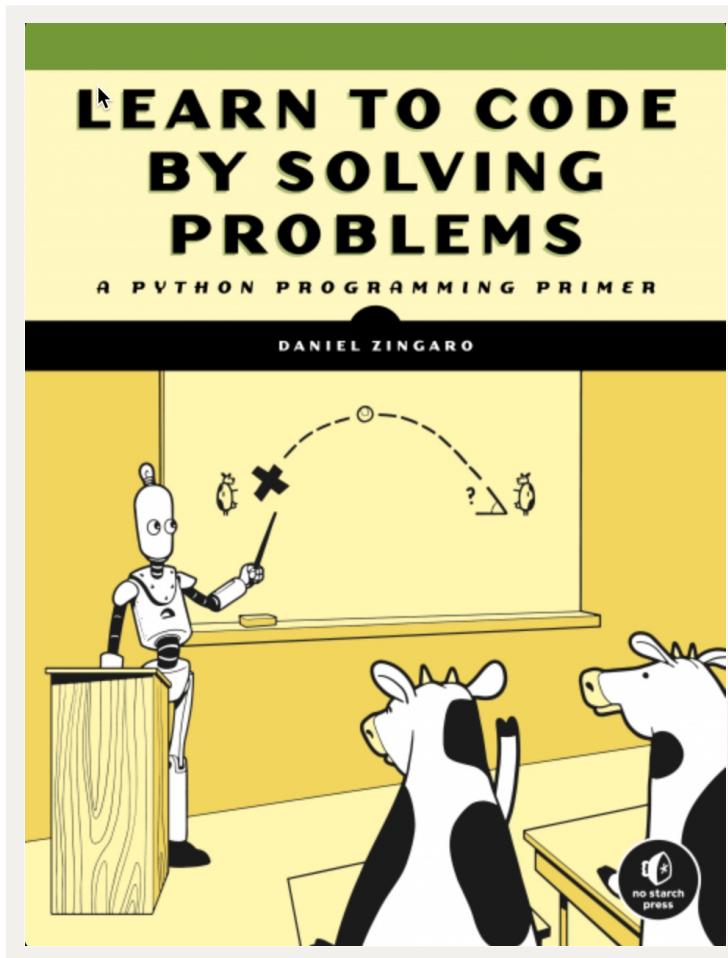
# Hoe leer je - door oefening - programmeren?

- Code analyseren: onderdelen en hun betekenis herkennen. Bestaande code begrijpen vooraleer je je eigen code schrijft
- Bouwstenenbenadering: analoog aan het leren van woordenschat, zelfstandige naamwoorden en werkwoorden alvorens zinnen te maken
- Eenvoudige onderdelen: kleinere problemen leren beheersen vooraleer de geleerde logica toe te passen op complexe uitdagingen
- Volledige systemen: projecten voor gevorderde studenten of meer gestuurd als onderdompeling waarbij concepten en syntax worden geïntroduceerd wanneer de oplossing voor het probleem hun toepassing vereist

# Hoe leer je - door oefening - programmeren?

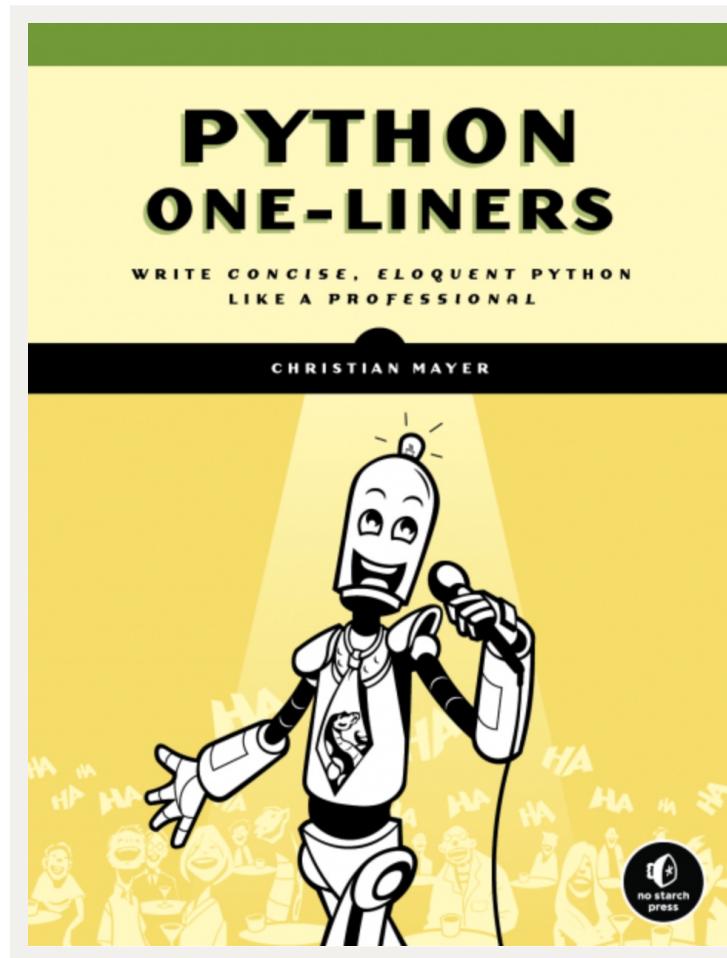
- Programmercompetentie kan het best worden bereikt door een gecombineerde aanpak
- Zo veel mogelijk op maat van de student
- Er zijn heel veel vormen van oefeningen mogelijk, gaande van heel gestructureerd en geleid tot volledig open projecten

# Diversiteit in oefenvormen



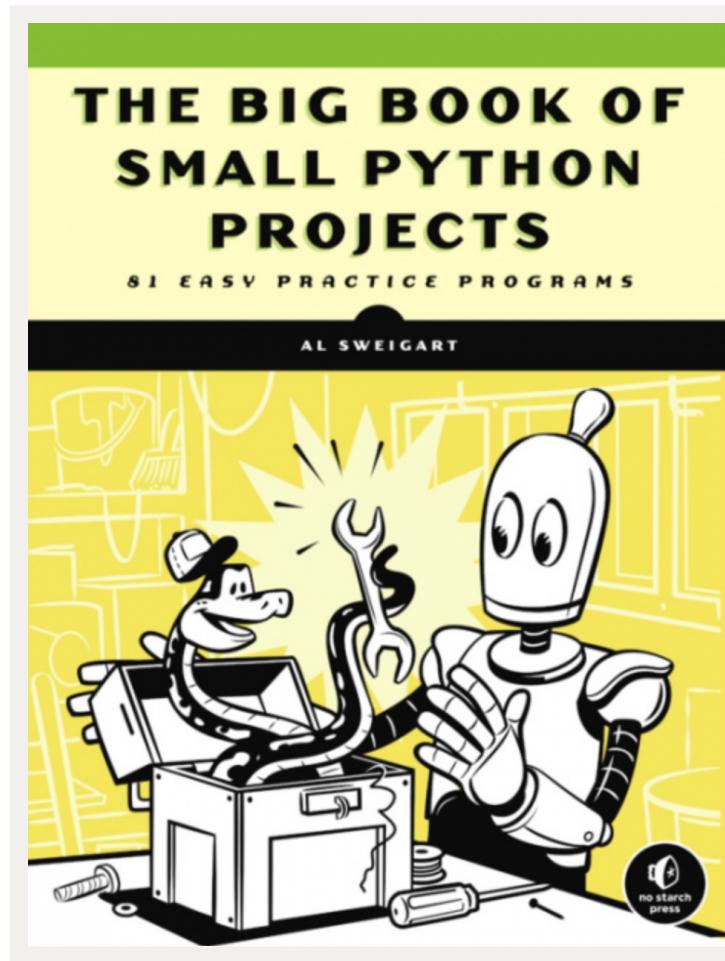
<https://nostarch.com/learn-code-solving-problems>

# Diversiteit in oefenvormen



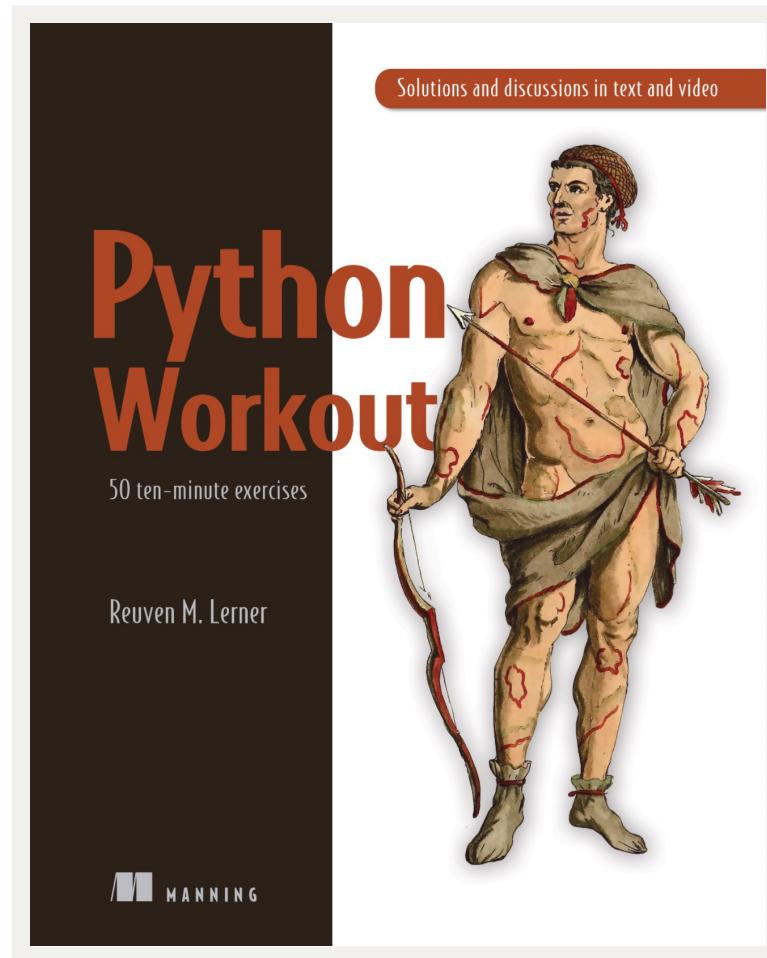
<https://nostarch.com/pythononeliners>

# Diversiteit in oefenvormen



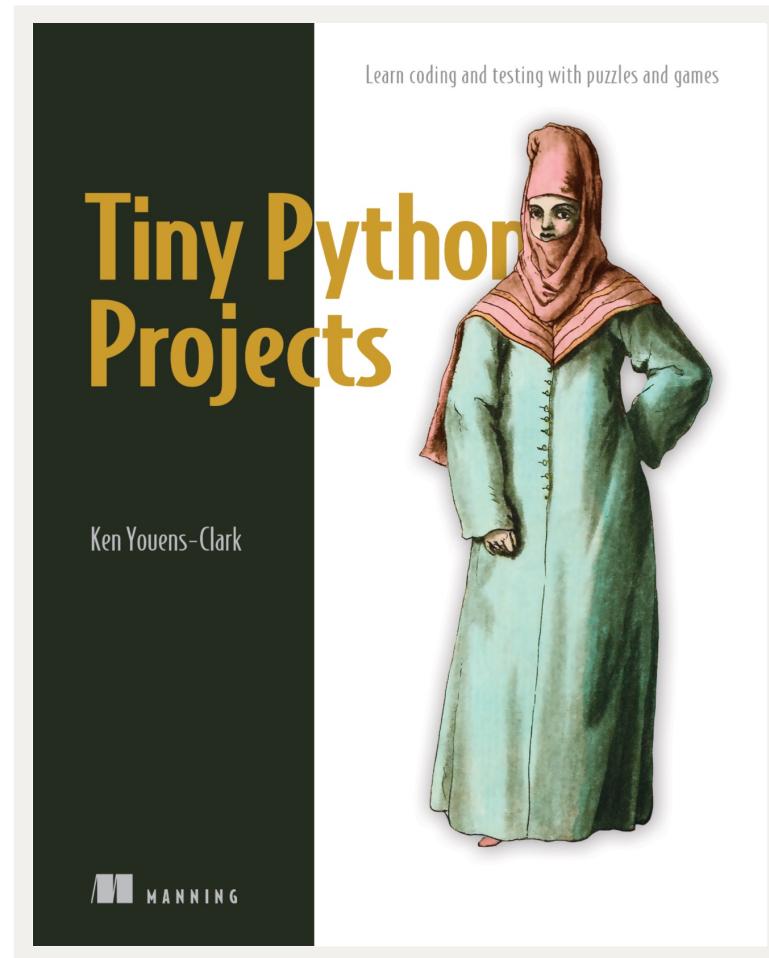
<https://nostarch.com/big-book-small-python-projects>

# Diversiteit in oefenvormen



<https://www.manning.com/books/python-workout>

# Diversiteit in oefenvormen



<https://www.manning.com/books/tiny-python-projects?query=python>

# Diversiteit in oefenvormen

 MANNING

search liveProject...

 browse

**liveProject**

## the best way to learn is by doing

Succeeding in tech takes years of experience. Often that means building software and working with tools that your day job might not cover. liveProject lets you **develop your full potential and advance your career much faster**.

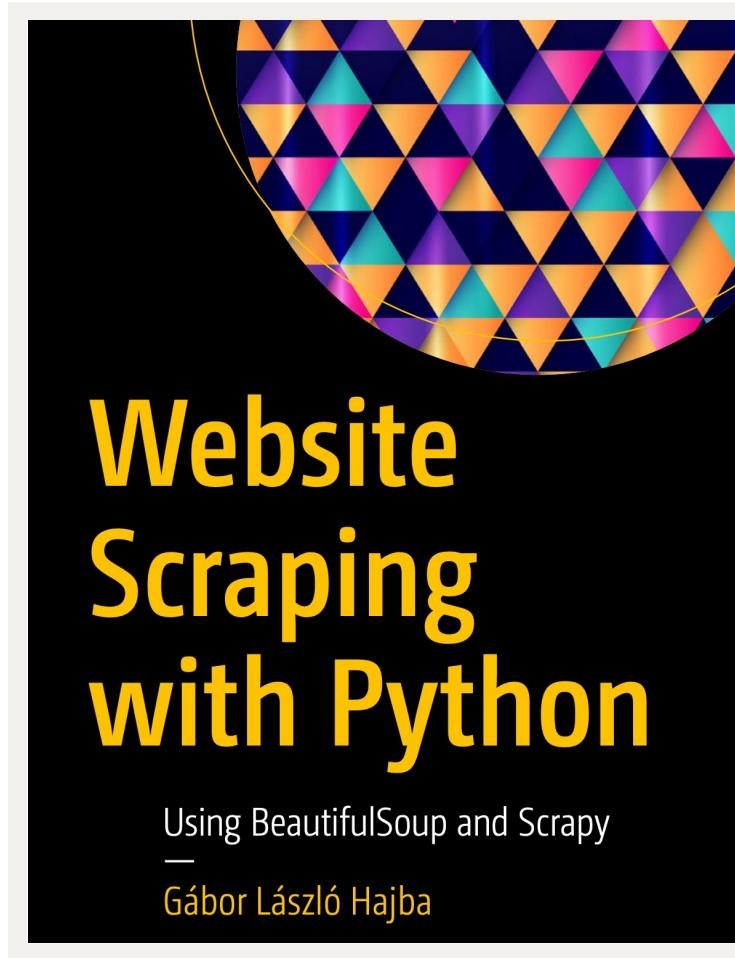
liveProject gives you the opportunity to learn new skills by completing real-world challenges in your local development environment. You'll be introduced to a variety of new technologies, gain access to the book and video resources you need to learn, and get just enough help to succeed. Solve practical problems, write working code, analyze real data — **with liveProject, you learn by doing**.

[Start a liveProject Today](#)



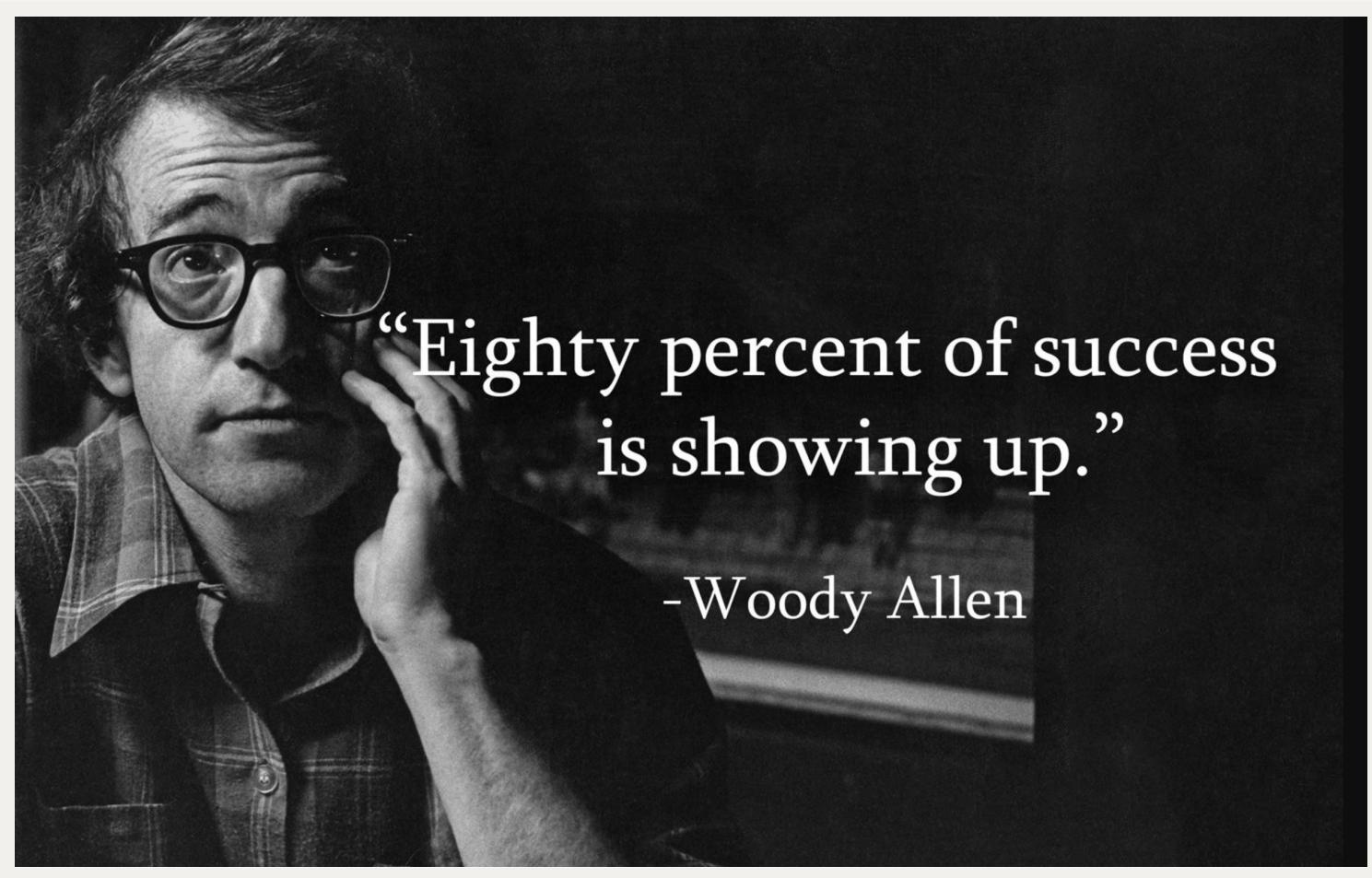
<https://liveproject.manning.com/>

# Diversiteit in oefenvormen



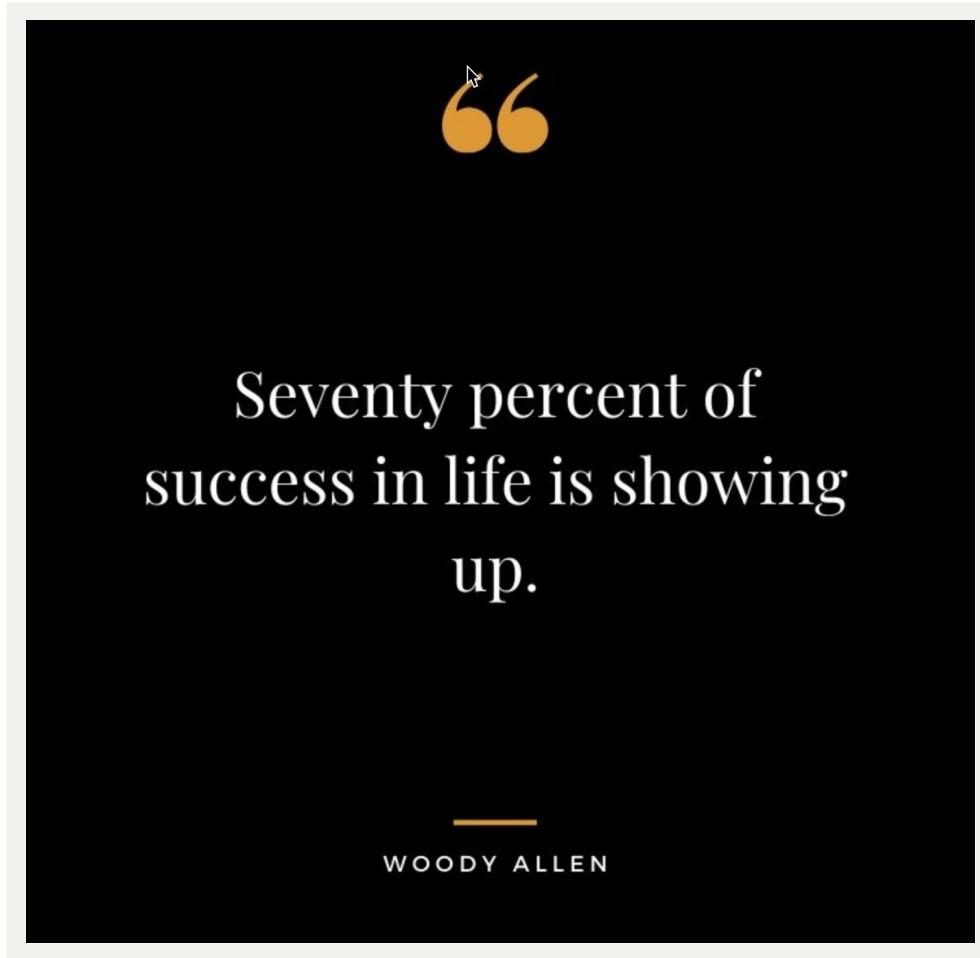
<https://www.apress.com/gp/book/9781484239247>

# Onthou!



Niet gewoon je broek verslijten: actief volhouden!

# Of was het 70%?



:-)

# Anyhow...



Werk is werk! Stap per stap! Met focus!

**Bedankt! - Discover IT - Leren programmeren?**

[kristof.michiels01@ap.be](mailto:kristof.michiels01@ap.be)