



12 REGLER

Lära sig koda

1. Lura hjärnan med 20-minutersregeln

- Tid?
- Människor har en tröghet, att inte göra en förändring.
- Koda “bara” 20 minuter. Plötsligt har det gått ett par timmar.
- En vana byggs med daglig repetition i en månad.
- Grupparbete: Hur får man motivation att göra detta?

2. Koda med ett syfte

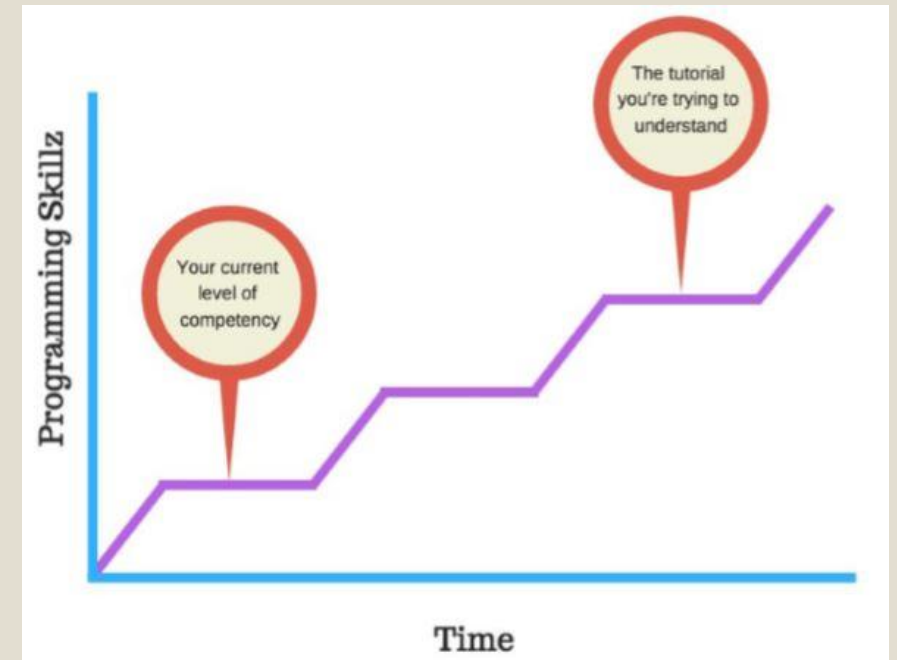
- Svårt att motivera sig att göra en app som listar primtal eller ojämna siffror.
- Om syftet med att lära sig koda, är att lära sig koda, blir man inte speciellt bra på det.
- Om syftet är att bygga en bra, smart eller nyttig app, så kommer du vilja bli bra på kodning.
- Logiskt tänkande, kreativitet.
- I början blir det kanske inget avancerad.
- Exempel:
 - App som väcker en minut tidigare varje dag.
 - En mors-dag-app, med slideshow.
 - Kalkylator för stektid för kött, baserat på vikt och tjocklek.
- **Grupparbete: Fundera ut minst två projekt, som du skulle vilja bygga.**

3. Det finns inget “perfekt programmeringsspråk” att lära sig

- Programmeringsspråket är bara ett verktyg.
- Det viktiga är vad du vill utföra med verktyget.
- Det mesta är samma: **loopar, villkor, variabler, funktioner** är samma i nästan alla språk.
- Skillnaden är mest syntaxen, orden. Varulv och Werewolf = samma sak.
- Swift: `print("Hello Werewolves")`
- Java: `println("Hello Varulv")`
- Grupparbete: Känner du dig trygg i de grundläggande programmeringsorden? Vad saknas?

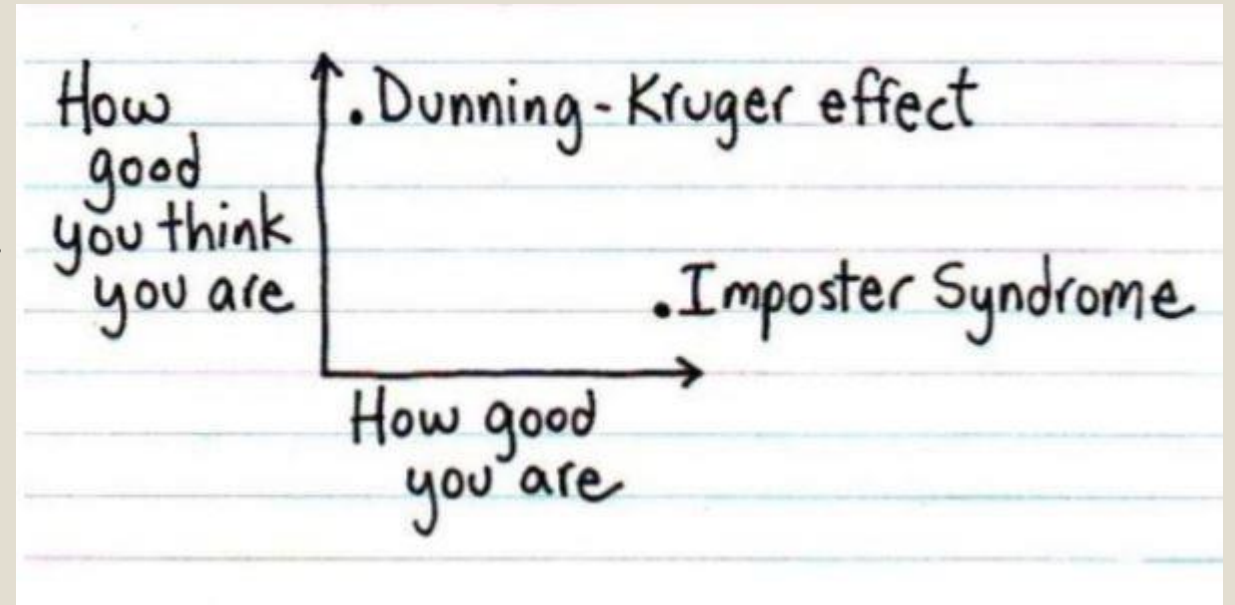
4. Förstå vad du skriver

- Följa en tutorial, eller code-along, men halvvägs så blir det en massa fel, som du inte har en aning hur du ska fixa.
- Problemet: Koda själv, utan hjälp, eller koda i en klass, med olika kunskapsnivå.
- Stegvis öka kunskapen
- För brant stigning, tappar bort dig
- För flack stigning, uttråkad.
- Om kunskapsnivån är fast, så kan du bara påverka tid.
- Ju mer nybörjare, desto mer tid behöver läggas.
- **Grupparbete: Behöver du lägga mer tid?**



5. Det är okej att inte veta

- Imposter Syndrome, känner sig som en bluff, och underskattar sin kompetens.
- 70% av mänskligheten har det.
- Anses man dålig om man googlar, eller går in på Stack Owerflow?
- Det är mer illa att inte kolla upp på nätet.
- Omöjligt att ha allt i huvudet.
- Kunskap är högt värderad i ett samhälle där information är svårt att få tag på. (1800-talet)
- Att kunna tänka, är högt värderat idag.
- Grupparbete: Var är du i diagrammet?



5. Det är okej att inte veta - forts

- - (id)initWithBitmapDataPlanes:(unsigned char **)planes pixelsWide:(NSInteger)width pixelsHigh:(NSInteger)height bitsPerSample:(NSInteger)bps samplesPerPixel:(NSInteger)spp hasAlpha:(BOOL)alpha isPlanar:(BOOL)isPlanar colorSpaceName:(NSString *)colorSpaceName bitmapFormat:(NSBitmapFormat)bitmapFormat bytesPerRow:(NSInteger)rowBytes bitsPerPixel:(NSInteger)pixelBits;
- Problem? Gör Såhär:
 - Börja med att fundera själv.
 - Det funkade innan, men sedan lade jag till de tre raderna kod.
 - Har jag missat några parenteser, citattecken, felstavningar eller semikolon?
 - Gå ifrån en stund.
 - Googla
 - Sök på felmeddelandet du fick, eller på det du försöker göra. (Skapa metod, defiera variable, instanisera objekt)
 - Gå in på StackOverflow (<https://stackoverflow.com/>)
 - Fråga kollega, lärare eller annan elev
- Kom ihåg: Det finns inte ett problem, som inte redan någon har haft.
- **Grupparbete: Börjar du känna igen de vanligaste problemen? Nämn några.**

6. Var en copycat

- Böcker och webben är bra referenser, men vill du lära dig: Bygg något själv!
- Saknar ideer? Var en copycat.
- Gör Notepad, MSPaint, ett piano, minesweeper, Tetris, Flappy Bird.
- Väljer du något som redan finns, är chansen stor att du kan få hjälp på nätet, om du kör fast.
- **Grupparbete: Fundera på appar/projekt du skulle vilja kopiera, och varför?**

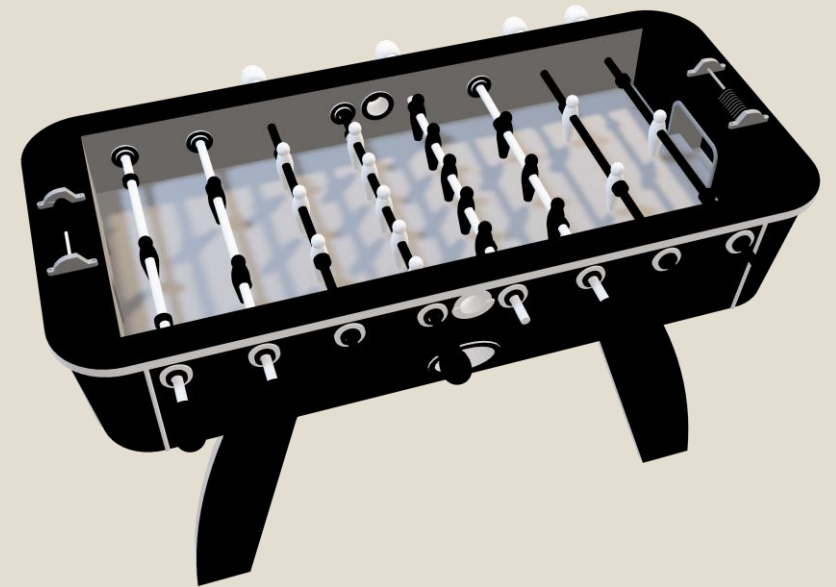
7. Var ansvarig

- Var ansvarig gentemot någon annan. Visa ditt arbete.
- Motivation är sällan stark i oss.
- Matcha ihop dig med någon. Ibland behöver du hjälp, ibland din “kodarkompis”.
- Kontroversiellt: Man värderar inte det som inte har ett värde (kostnad).
- Bara det man mäter kan förbättras.
- Grupparbete: Diskutera om ni skulle vilja vara kodarkompisar.

8. Fortsätt lära dig

- För att fortsätta vara relevant, uppfinn dig själv på nytt.
- Nya trender, teknologier, ramverk, spark
- Var inte en CD-drive, hörlursuttag eller modem.
- <https://learnxinyminutes.com/>
- Grupparbete: Vad skulle du vilja lära dig mer om?

9. Spela Foosball



- På film så skildras programmeraren som någon som skriver rad på rad kod, snabbt.
- I verkligheten: En programmerare stirrer mest på skärmen.
- Det finns alltid en bug, eller ännu värre, något som fungerar fast det inte borde.
- Har du ett problem?
 - Gå ifrån koden
 - Sov på saken
 - Spela Foosball
 - Ta en promenad
- I 9 fall av 10 kommer lösningen visa sig. I sista fallet, be om hjälp.
- Koda mindre, tänk mer. Den enklaste koden att städa bort, är den som aldrig skrevs.
- **Grupparbete: Hur vill du göra när du vill gå ifrån koden?**

10. Skaffa en mentor

- Lärarens roll
- Bekanta, folk på nätet, klassen
- Par-programmering. En som skriver koden (learner) och en som tittar och bedömer koden (mentor).
- Kan upplevas obekvämt
- Du får år av kunskap på några timmar.
- En bra mentor löser inte dina problem, utan ställer frågor som får dig att tänka själv.
- Häng inte upp dig på information, lär dig tänka.
- <https://www.meetup.com/>
- Erbjud hjälp tillbaka, kanske om något helt annat.
- **Grupparbete: Var kan man hitta mentorer? Finns några i klassen?**

11. Dela upp ditt app-projekt

- Du har en fantastisk appidé, men den är för stor, svår och komplicerad. Dela upp den.
- Rostad-bröd-robot:
 - Roboten behöver inte förstå vad rostat bröd och smör är.
 - Tre steg:
 - Placera brödet
 - Hämta en klick smör med en kniv
 - Bre ut smöret på smörgåsen
 - Frågor:
 - behöver roboten bre smöret, eller räcker det att det smälter ut på smörgåsen
 - Måste roboten plocka upp kniven, eller är den inbyggd i armen.
- Ju mer du bryter ner problemet, desto enklare att hantera varje delproblem.
- **Grupparbete: Hur skulle du dela upp en projekt, såsom en självkörande bil?**

12. Bryt ner någon annans kod.

- En grundregel är: Kopiera inte kod du inte begriper.
- En metod att hantera detta:
 - Kopiera kod från t ex StackOverflow, eller en hel app från Github
 - Kör din app, och se till att den fungerar.
 - Ta bort den inklistrade koden, rad för rad
 - Vilka felmeddelande får du?
 - Fungerar koden?
 - Vad hände med din app när du tog bort raden?
 - Ta även bort rader du begriper, för att bekräfta att du hade rätt.
 - Byt plats på rader.
 - Kan det fungera med en annan ordning?
 - Varför är koden skriven i den här ordningen?
- Reverse Engineering:
 - Ladda ner någon app. Se hur den fungerar.
 - Sedan bygger du den själv från början.
 - När du är klar, jämför koden.
- **Grupparbete: Skulle ovanstående metod fungera för dig? Finns det andra sätt?**

Övning

- Grupparbete, 2-3 personer i varje grupp
- Gå igenom de 12 reglerna.
- Diskutera om det är något du gör idag?
 - Fundera på hur man kan uppnå detta och vad krävs?
 - Diskutera de specifika frågorna, till varje regel (Rödmärkta)
- Sammanställ era slutsatser och projektförslag på max ett A4.