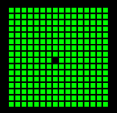
How do we make code ***fun*** and more ***intuitive***?

2017-11-08, Øredev Developer Conference @ Malmö, Sweden
Room: Homo Erectus

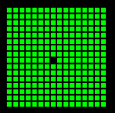


Overview

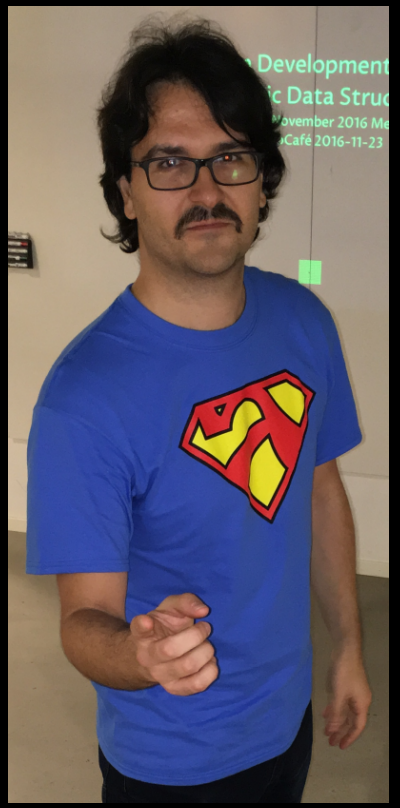
- About me (very shortly)
- Background:
  - Coding Pirates
  - Scratch
- Moving on from “picture books” to “real books”
  - Microsoft to the rescue !!!
  - Demo + show me some code

Note: Slides are released under the CC BY-SA license

- Creative Commons Attribution-ShareAlike (“copyleft”)



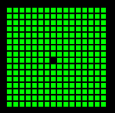
About me (very shortly)



- Ramón Soto Mathiesen (Spaniard + Dane)
- MSc. Computer Science ***DIKU/Pisa*** and minors in Mathematics ***HCØ***
- ***CompSci*** @ SPISE MISU ApS
  - ***"S**tay **P**ure, **I**solating **S**ide-**E**ffects"*** -- Michael Werk Ravnsmed dixit
  - ***"M**ake **I**llegal **S**tates **U**nrepresentable"*** -- Yaron Minsky dixit
  - Trying to solve EU GDPR with a scientific approach (Computer Science and Math)
  - ***Elm*** (***JS*** due to ports) with a bit of ***Haskell*** and a bit of ***F#*** (fast prototyping)
- Elm / Haskell / TypeScript / F# / OCaml / Lisp / C++ / C# / JavaScript
- Founder of Meetup for F#unctional Copenhageners (MF#K)
- Volunteer at Coding Pirates (Captain at Valby Vigerslev Library Department)
- Blog: http://blog.stermon.com/ and Twitter: @genTauro42

Open Source .NET projects

- Previous workplace (CTO of CRM @ Delegate A/S):
  - MS CRM Tools:
    - http://delegateas.github.io/
  - ~~Delegate.Sandbox~~, isolated side-effects at run-time in F#
    - http://delegateas.github.io/Delegate.Sandbox/
- Current workplace (SPISE MISU ApS):
  - Syntactic Versioning (SynVer @ F# Community Projects)
    - Mostly driven by swede Oskar Gewalli, @ozzymcduff
  - Puritas, isolated side-effects at compile-time in F#
    - F# eXchange 2017 (talk and video)



Background: Coding Pirates

- Coding Pirates is a non-profit organization that tries to help kids understand technology so they are able to create and not just consume
- There are many Departments in Denmark
  - https://codingpirates.dk/afdelinger/
- In Copenhagen, we organize Hackathons every 2 months in order to help kids that don't have the possibility to participate regularly at the Coding Pirates departments, due to lack of volunteers
  - Usually all tickets are purchased on a single day, just like music concerts !!!

Background: Coding Pirates

Girls coding @ Hackathon, Hvidovre

Background: Scratch

- First time I began to use it was when I became a volunteer at Coding Pirates
- Is a free and Open Source visual programming language developed by the MIT Media Lab
  - Online client:
    - Stable (Adobe Air): https://scratch.mit.edu/
    - Beta (HTML5): https://llk.github.io/scratch-gui/
  - Scratch @ GitHub: https://github.com/LLK
- Easy to get started (***simple***, ***fun*** and ***intuitive***)
- Here are a few other features of the language ...

Scratch Features: Composition

- Focus on composition in order to increase creativity

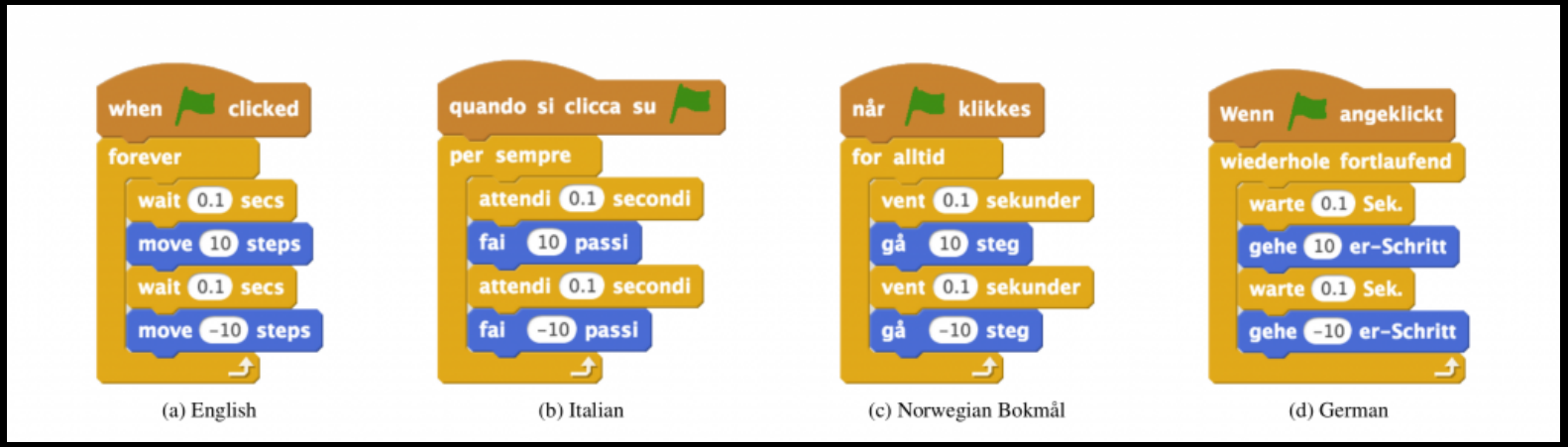
Scratch Features: Strongly type-safe

- You can't write wrong code and then execute it

Scratch Features: Built-in REDL (REPL)

- Read, Evaluate, Draw (Print) and Loop

Scratch Features: Translation support (like Excel)

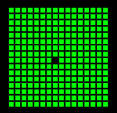


(a) English (b) Italian (c) Norwegian Bokmål (d) German

- Possibility to set the language for the Project Editor and code blocks in more than 40 languages:
  - https://wiki.scratch.mit.edu/wiki/How_to_Translate_Scratch

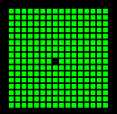
Scratch Summarized

- ***Simple***, ***fun*** and ***intuitive***
- Composition
- Strongly type-safe
- Built-in REDL (REPL)
- Translation support (like Excel)



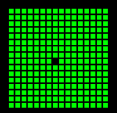
Moving from “picture books” to “real books”

- Sadly this awesome programming language has a major flaw:
  - Older kids tend to find it boring after a couple of seasons
  - I tend to say that coding in ***Scratch*** is like reading “***picture books***” while coding with ***text based programming languages*** is like reading “***real books***”
- So, the question is:
  - How do we move-on from ***Scratch*** to a text based programming language keeping it ***simple***, ***fun*** and ***intuitive*** while maintaining all the fancy features we just described?



Moving from “picture books” to “real books”

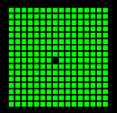
- Microsoft to the rescue !!!
- With a few ***Microsoft Tools*** and ***Cloud Services***, we will be able to provide the next step in the learning path for the children :)
- Lets begin



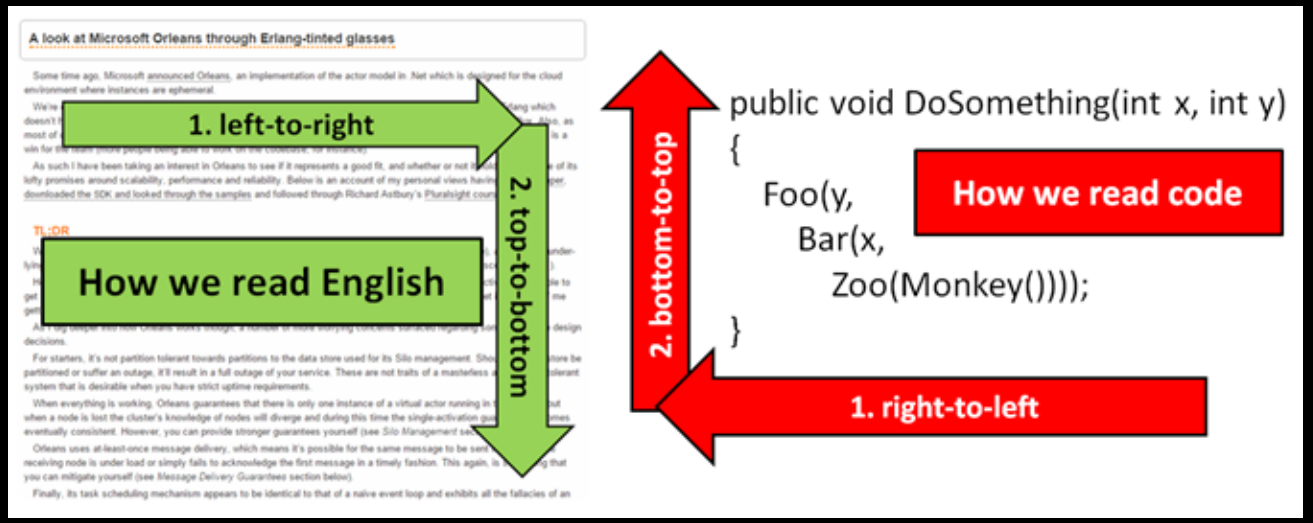
Microsoft to the rescue !!!
The main tool (F#)

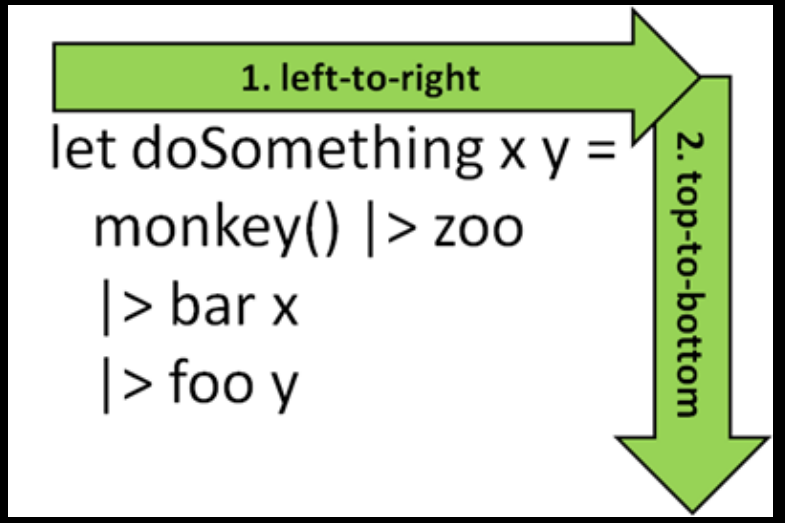
- We need to find a programming language that is:
  - ***Simple***, ***fun*** and ***intuitive***
  - Support for Composition
  - Strongly type-safe
  - With a built-in REPL
  - Translation support (like Excel)

  Note: Python is a valid choice, but it lacks of type-safety and therefore no compiler can help the kids (it doesn't scale). OO languages require to much noise in order to write a single line of code and they lack of scripting (REPL) features



Microsoft to the rescue !!!
Simple, ***fun*** and ***intuitive***



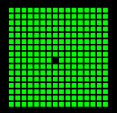


- Forced indentation, just like Python, in combination with |> operator, makes it easy to read again & again

Microsoft to the rescue !!!
Support for Composition

- Higher-order functions.
  - Passing functions as arguments is just like gluing functions together, composing them to bigger ones:

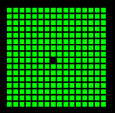
```
[0 .. 9] |> List.map (fun x -> x + x)
```

Microsoft to the rescue !!!
Strongly type-safe

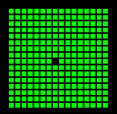
- Computer says no :

```
(* error FS0001: The type 'string' does not match the type 'int' *)
let result = 42 + "42"
```

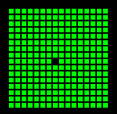
Microsoft to the rescue !!!
With a built-in REPL

- Read, Evaluate, Print and Loop (REPL):
  - Possible to evaluate functions, modules and types directly from the IDE to F# interactive (interpreted code)
  - This makes it easy to reason about creating smaller pieces of logic and composing them to greater blocks
  - F# script files (.FSX) are also interpreted, which means that files are type checked before executing a single line



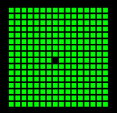
Microsoft to the rescue !!!
Translation support (like Excel)

- Hmmm, F# doesn't really has this feature though ...
- But wait a second. F# is a really simple language and by using a few basic keywords (`let`, `fun` or `function` and `module`) and some operators, you can actually go pretty far
- Let me explain



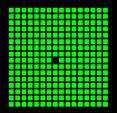
Microsoft to the rescue !!!
Translation support (like Excel)

- By using the Microsoft Azure:
  - Cognitive Services Translator Text API
- We can easily write some code, in order to translate English terms to any other language supported by the service:
  - Snippet hosted at my blog:
    - F# - Cognitive Services Translator Text API
- In combination with my ***Open Source .NET*** project, SpiseMisu.SemanticVersioning, we should easily translate parts of F# Core, specific modules, in order to ease the language for kids



Microsoft to the rescue !!!
Translation support (like Excel)

- Sadly, the naming isn’t consistent enough, which makes it almost impossible to produce decent results :(



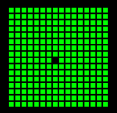
Microsoft to the rescue !!!
Translation support (like Excel)

```
module Liste =
  type 'a liste = 'a list

  let mappe :
    ('vilkårligFra -> 'vilkårligTil)
    -> 'vilkårligFra liste
    -> 'vilkårligTil liste =
      List.map

  let reducere :
    ('vilkårlig -> 'vilkårlig -> 'vilkårlig)
    -> 'vilkårlig liste
    -> 'vilkårlig =
      List.reduce
```

- Expected result (Danish naming)



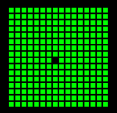
Microsoft to the rescue !!!
Translation support (like Excel)

```
[<RequireQualifiedAccess>]
module Dansk =
  [<RequireQualifiedAccess>]
  module List =
    let append
      :  list1:('T) list
      -> list2:('T) list
      -> ('T) list
      =  Microsoft.FSharp.Collections.List.append

    ...
    (* The field, constructor or member 'forAll' is not defined. Maybe you want one of the following: forall or forall2 *)
    let forAll
      :  predicate:('T -> System.Boolean)
      -> list:('T) list
      -> System.Boolean
      =  Microsoft.FSharp.Collections.List.forAll

    ...
```

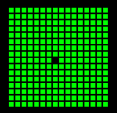
- Result w/o translation cos inconsistent naming



Microsoft to the rescue !!!
Translation support (like Excel)

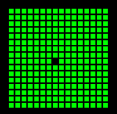
- So even though we aren't capable of translating the Core components of the language, we discovered that the older kids who get bored, have already learned enough English to code in simple languages
- Anglo-Saxon children really have an advantage when it comes to code in early ages as most programming languages have English keywords
  - Yaron Minsky, @yminsky, daughter coding Tic-Tac-Toe in Racket: tweet

Shout-out: To Simon J.K. Pedersen, @simped, for providing insight and helping out with the ***Microsoft Azure Services***. Kudos !!!



Microsoft to the rescue !!!
Lets recap

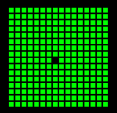
- We need to find a programming language that is:
  - ☑ ***Simple***, ***fun*** and ***intuitive***
  - ☑ Support for Composition
  - ☑ Strongly type-safe
  - ☑ With a built-in REPL
  - ☒ Translation support (like Excel)
- But is this enough? Apparently it isn't. You still need to go the extra mile in order to catch the kids attention … So what do all kids like?



Microsoft to the rescue !!!
Minecraft

- All kids ***love*** to play Minecraft (bought by MS, therefore a ***MS Cloud Service***)
- For us it was LEGO, even though we were limited to build based on the amount of blocks that our parents could purchase, while for them, they have infinite amount of blocks to build whatever they want in order to express their creativity
- So knowing this, is there something we could do in order to combine a ***simple***, ***fun*** and ***intuitive*** text based programming language and this game?

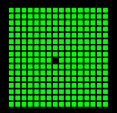
  Note: There is a reason we don't handle the username and passwords to the kids, as they would be playing instead of trying to learn something else. Yeah, we all did it as kids right? Raise you hand if you didn't, so I can point out a liar !!!



Microsoft to the rescue !!! Minecraft

- The answer is yes of course !!!
- Chris Dobson did write the following blog post for the yearly F# Advent Calendar 2015:
  - F#, Minecraft and a Raspberry Pi , but it turned out, that this was only usable with Raspberry Pi ***Minecraft*** clients, as the Console App was ***hooking up*** to the ***Minecraft*** client itself and not to a server
- So there was no other thing to do than code the client from ***Scratch***, pun intended. A few honorable mentions:

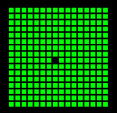
  Note: And what better place than the land of ***Mojang*** to showcase it !!!



Microsoft to the rescue !!!
Minecraft → Protocol

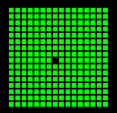
- Implementing the very well documented Protocol
  - Data types (VarInt, Chat, ByteArray, ...)
  - What's the normal login sequence for a client?
  - Chat Commands
  - Minecraft Blocks and Colors
  - Around 300 LOC (Lines Of Code)

Shout-out: To Pokechu22 from #MCDevs for whom and without his help, I might not had been able to create this client. Kudos !!!



Microsoft to the rescue !!!
Minecraft → TCP Client

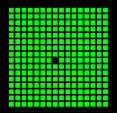
- Implementing the TCP Client
  - Unix / Berkeley sockets
  - Minecraft Data Package Format
  - Async stream (I even found a bug in AsyncRead → tweet)
  - Around 400 LOC (Lines Of Code)



Microsoft to the rescue !!!
Minecraft → DSL

- DSL (Domain Specific Language) for ease the creativity and avoid troubleshooting code
  - Easy to define with F# Computation Expressions. You can think of them as LINQ for C# but instead of only working for data collections, you can create your own custom ones
  - Only implemented `teleport` and `fill` as they are what is mostly needed in order to build
  - Just like with ***Scratch***, kids can't write erroneous code and then execute it. Computer says no
  - Around 50 LOC (Lines Of Code)

Note: F# now has Elm alike compiler error messages with suggestions

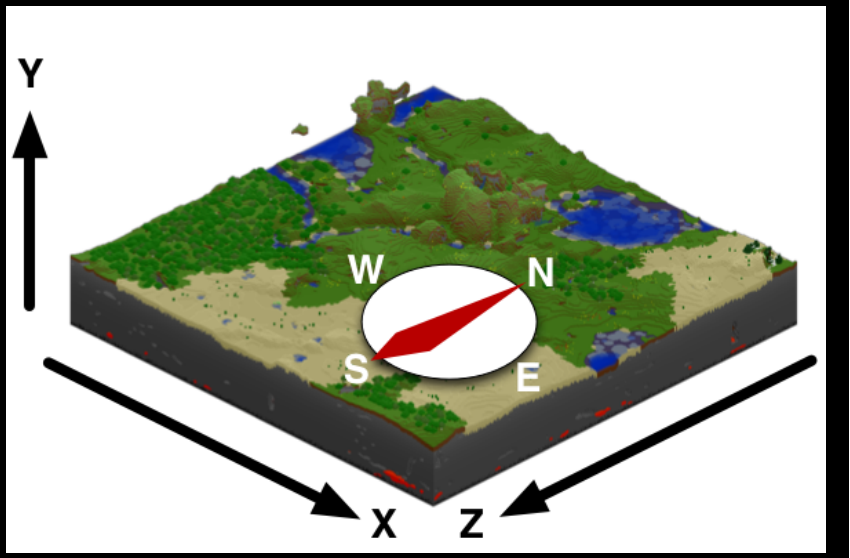


Microsoft to the rescue !!!
Minecraft → DSL

```
module Unsafe =

  (* TP to origo facing +X *)
  let origo : Data.Packet list =
    [ Data.packet 0x02
        [ "Message" => Data.string "/tp 0 4 0 270 0"
        ]
    ]

  let foundation =
    [ Data.packet 0x02
        [ "Message" => Data.string "/fill ~50 3 ~ ~ 3 ~50 concrete 8"
        ]
    ]
```



- Writing ***Minecraft*** Commands unsafely (text)

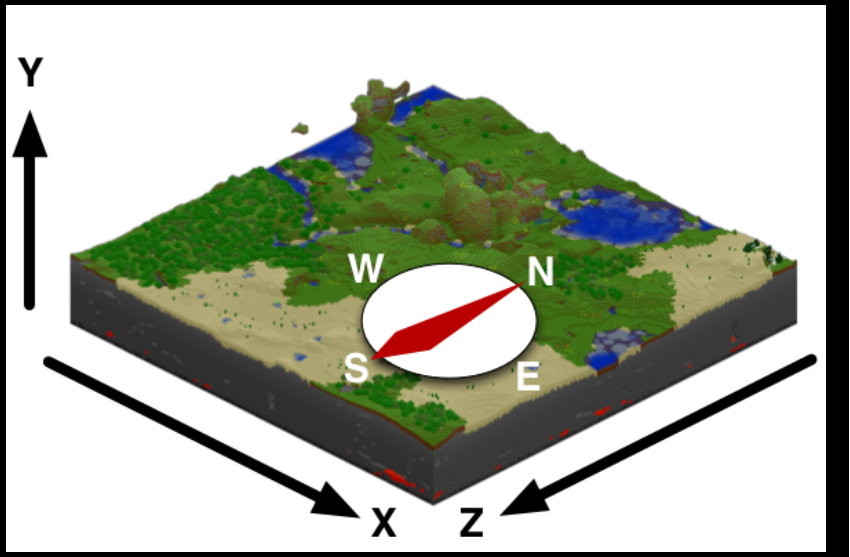
Microsoft to the rescue !!!
Minecraft → DSL

```
module Safe =

  let pilar : Data.Packet list = minecraft {
    fill
      Relative 0 Relative 0 Relative 0
      Relative 0 Relative 4 Relative 0
      Concrete White
  }

  let ceil : Data.Packet list = minecraft {
    fill
      Relative -1 Relative -1 Relative -1
      Relative 49 Relative -1 Relative 49
      Concrete White

    teleport
      Relative -1 Relative 0 Relative -1
      East Horizon
  }
```



- Writing ***Minecraft*** Commands safely (DSL)

Microsoft to the rescue !!!
Minecraft

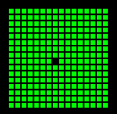
- Demo + show me some code

Summary (1/2)

- With a few of ***Microsoft Tools*** & ***Cloud services***, we are able to make code ***fun*** and more ***intuitive*** for children when they transition from ***Scratch*** to ***text based languages***:
  - F# also have a major flaw which is that it's not really a ***cross platform language***. It works really great on Windows, but it still ***lacks a bit*** on ***Mac*** and ***mostly Linux***, both the language itself as well as in tooling
  - For that reason, one of the other volunteers from my CP Department, ***Hans Bugge Grathwohl***, now teaches text based programming with Elm + Ellie (The Elm Live Editor), which is fine, as it also a very ***simple***, ***fun*** and ***intuitive*** ML language (***Meta Language*** and not ***Machine Learning***) that carry on the nice features mentioned until now (except translation ofc)

Summary (2/2)

- I will write a blog post on the 6th of December for the F# Advent Calendar 2017 with regard of the ***Minecraft*** client, and by then I will release the code under a ***free*** and ***Open Source License*** (OSL)
- ***"If us who can, don't go the extra mile, who will then do it?"***
  - Think about that. If you think you can make a difference, please join a similar organization in your country, so we can teach kids how to ***code***, and maybe even some ***Computer Science***, in a ***simple***, ***fun*** and ***intuitive*** way
  - We don't expect everybody to write ***Minecraft*** clients from ***Scratch***, again pun intended, or something similar. But if you can, please do. **Remark**: There is room for all of us volunteers :)
- ***Final thought***, if we can teach kids to code easily with ***simple***, ***fun*** and ***intuitive*** text based programming languages, shouldn't we also be able to teach grown ups as well?



Q & A

Any Questions (now) or later @genTauro42?