A Software Engineer That Inspires Me:
Mitchel Resnick

"When you grow up you tend to get told that the world is the way it is and your life is just to live your life inside the world. Try not to bash into the walls too much . Try to have a nice family life, have fun- save a little money. But that's a very limited life. Life can be much broader, once you discover one simple fact and that is that everything around you, that you call life was made up by people that were no smarter than you. And you can change it, you can influence it. You can build your own things that other people can use".

I remember the first time I saw Steve Jobs say those words. I was sixteen at the compute TY module , a class put on by DCU to get transition year students interested in computer programming. At the time , my head was very much undecided whether I wanted a career in physics or law, but definitely one of those two things. The compute TY module was free and it just so happened there was no football or basketball on that day, so this was the only remaining alternative to miss a day of school. We listened to videos of Willi.am and several other celebrities tell us about the importance of programming. In all honesty though, up until this point it had all been drivel. It's all well and good Will.i.am or Adam Lavigne telling me I should be a computer programmer , but as I couldn't really imagine them hunched over a bread board or awake at 2am debugging code- to a degree it just felt quite scripted.

Then jobs came on screen. In what was clearly older footage, with distinct home video vibes, he pledged to give me some life advice. Not knowing much more about him than his bank balance I decided it might be a good idea to take note of what he was going to say. I did not expect what he was going to say would change my life. But it did. I think everyone gets to a point in their life where they just concede mediocrity. You envisage yourself on the bell curve that is life to be somewhere about the median in every facet of your being. And yet still, here was someone who I considered to be the smartest man on earth telling me that I was as smart as him. Telling me that just as he had built things, I too could build things, things that had never existed before. I could make them. I don't believe Jobs' was solely addressing the small number of people with the capacity to forge as successful a company as Apple or make technology as earth shattering as Facebook. I think he speaks to the innate creativity of everyone. I remember another video being played after but I couldn't even tell you what it was about, that's how deeply Jobs' words resonated with me.

After that next uninspiring video it was time to get my first ever taste of programming. It was all well and good Steve Jobs telling me I could build things for other people to use. Just in my case they might not necessarily be the *same type of things* that Jobs may have had in mind. Still to this day, I have qualms with the word *"coding"* . Whenever I tell someone I am a computer science student, the next question is invariably "*oh is that just like a load of coding and stuff?*". When you think of the word code the first thing to spring to mind is coded messages, ciphers – things which are intrinsically difficult to understand . The term "*coding*" is as ambiguous and daunting as Hollywood would like to make it seem, with their very active terminal windows and constant references to algorithms and encryptions. Even if it paid well, a life of being hunched over a computer screen trying to decipher and write endless sequences of "*code"* into a terminal window was in my view no life at all. As far as I was concerned "*coding*" was just not for me.

A Software Engineer That Inspires Me:
Mitchel Resnick

But still, as I was at the course so I was obliged to give it a go. Although I clearly had some strong biases about coding, I am still open to trying new things. The course I had signed up to do was called the "app inventor" course and was one of three modules you could do during the computer TY course. As such the course director instructed us all to open scratch on our machines as this would be the development environment we would be using. I think at the time I was so bad with that I needed help just to locate the programme on my desktop. But once I found that scratch icon everything changed.

"Today we are going to be building space invaders using scratch". We were given a handout with some basic project specs on it as well as some tips for how to get started with scratch. The interface was very intuitive and I quickly started making "stuff" happen on the screen. Initially I wasn't very good, but the guy beside me seemed to know what he was about and with his help it wasn't long before the "stuff" happening on my screen started to look more and more like space invaders. There was just one thing that really confused me. Where was the code? I was coding but everything I wrote was either just English or very basic maths. Where was the code? There wasn't any algorithms or encryption , no terminal window and yet I had made space invaders. Not only had I made space invaders, I had made my own version of space invaders. The planet being defended was Mars, the invaders were astronauts from NASA. Enemy Health and firing rate were variable depending on your progress in the game. This wasn't code, this was art. Everyone in the room had made space invaders but this was *my* space invaders. It just felt so personal to have created something. To have brought my own little ideas into the world. If this was code than code *was* art.

I actually could not believe it, that the concept "coding" which had before seemed so megalithic, complex and mysterious was actually nothing more than logical statements wrapped in English and executed sequentially. Normally Hollywood presents the viewer with an unrealistically simplistic view of life, where everything works out in the end and everyone's happy. In this case however, the reality was so much more simplistic than I had perceived. Using Scratch made me realise that Jobs was right, making my own things was a very realistic option for me. It's ultimately not Steve Jobs I have to thank for this however. It is in fact Mitchel Resnick – director of the Lifelong Kindergarten group at MIT, a group which has curated and overseen the development of the Scratch programming language.

It is quite amusing to think now, as a computer scientist when all my friends are discussing the benefits of R, Rust or React, that the language that has had the greatest impact on my life has been without a shadow of a doubt – Scratch. But it's so true , without the work of Mitchel Resnick and his team at MIT, I simply would not be who I am today. I sense this is the case for many students and professionals across the country. There is no programming syllabus or class taught in second level schools in Ireland. There are no TD's in our government with a background in technology. Despite the massive volume of tech giants in Ireland , there seems to be no urgency about educating people in Ireland about coding, what it is and how to do it. It is projects like Scratch that push back against the ignorance of those in power and empowers people to take their education into their own hands.

Since its's initial release in 2007, Scratch has been used to produce thirty four million projects from thirty two million unique users. That's thirty two million people that have gone on a journey where they will surely discover that the world of coding is not as scary as it's made

out to be. Mitchel Resnick , talking about the Kindergarten approach to learning cites what he calls the “four P’s” : projects, passion, peers and play. These are the driving factors and core beliefs of the scratch project he says. That users should be able to “make projects on their computers, based on their passions in collaboration with their peers and a playful spirit”. Coding Resnick argues, is analogous to writing; kids and adults both write in the same language, you don’t teach kids to write in a different language to that which adults write in. Why? Because the motivation behind writing is more than just the mastery of a skill, it is about an expression of ideas. We are in an age of technological advancement today as the world has never seen before. Coding is as important a skill as reading was during the renaissance. Whether you are going to be a professional programmer or not Resnick believes as I do too, that you should be able to express your ideas. Increasingly in this age of technological revolution that means being able to code.

If it’s not amazingly self-evident why Mitchel Resnick has earned my respect let me elaborate. Scratch is an educational tool. It was not made to generate massive revenues for Resnick and his team and yet with thirty two million users he could so easily have licensed the software out to schools to use for a small fee and made himself millions. But no, Resnick has completely reimagined the entire discipline of computer programming and software engineering. Children of the future will associate programming with the colourful feline face of the scratch logo and not the horrifically daunting images of terminal windows that I was predisposed to. Even for teens and adults who have not previously explored the world of computer programming, it couldn’t be made more accessible than it has been by the Scratch programming language. There is no longer a requirement from IBM, Apple or Google for you to hold a degree in order to work at these companies. If you were so inclined you could educate yourself using free online resources and potentially move on to work professionally as a software developer with absolutely no financial overhead cost. That is the platform that Resnick has created.

Although he will possibly never be credited with it, Resnick will be responsible for an entire generation of talented young software engineers. It is his vision, his product which will have kindled the flame of interest in so many engineers. He will by extension, be responsible to a certain degree for all of the projects and products produced by these engineers. Over the next fifty years how many major technological break throughs will be made by engineers who have had their first taste of programming from a Scratch environment?? I don’t think it is unreasonable to suggest that Mitchel Resnick and his team may have reshaped the face of technology on earth. By introducing talented young people to the world of software engineering in a way that is neither condescending nor boring and which encourages peer engagement and the creative thinking necessary to be a great engineer, Resnick has undoubtedly captured some great minds which would otherwise find themselves practicing law or medicine. If this does not bring about advancements in technology that would not otherwise have happened, it will at the very least increase the pace that technology is currently advancing at. As I say however, I suspect history may forget just how monumental Resnick’s contribution to computer science has been. Personally I think that is how he would want it though. I admire him very much as an educator and as a man. I don’t think he has set out for fame and fortune with this particular project. His goal as I see it is to change lives, and in my case he has certainly succeeded.