

IMPROVING AS DEVELOPERS



Belén Albeza


@ladybenko
belenalbeza.com

A BIT ABOUT MYSELF...

- 8 years of experience in the industry, in UK and in Spain
 - Game development for mobile and Facebook
 - Front-end, back-end, a bit of devops...
 - R&D and tiger team
 - DevRel at Mozilla
- First joined a developers community at 14 yo.



BEING A FRONT-END DEV TODAY




- LOTS of shiny new stuff every day
 - New frameworks, new tools, new API's, new CSS properties...
- LOTS of meet ups and conferences to go to
- LOTS of pressure to "contribute to the community"
 - Do you have projects on Github?
 - Do you contribute to open source?
 - Do you write technical articles?

WHO IS OVERWHELMED?

VAMO A CALMARNNO



IN THIS TALK

- 
- We'll see which myths & beliefs are holding us back
 - How we can improve our performance, and learn effectively during our career
 - How we can make our teammates and colleagues better

MYTHS & BELIEFS

PASSION

ARE WE SURE “PASSION” IS WHAT WE ARE LOOKING FOR?



- “Passion” tells nothing about a person’s talent, skills, or performance
- “Passion” is sometimes used as an excuse to get advantage of people
- Remember that we can be passionate and still have a life outside software
- Let’s look for **professionalism!**




Professionalism is about knowing your job, doing it well, and being proud of it [...]

Passion is no guarantee of talent or even basic competence. Ability, pride, discipline, integrity, dedication, organization, communication, and social skills are much more useful to an employer than passion is.

- Ernest Adams
Passion versus Professionalism (Gamasutra)

OUR HEROES

BAD ROLE MODELS

- 
- Questionable ethics / behaviour
 - Being a genius doesn't excuse bad actions
 - It's naïve to assume their beliefs / actions don't affect their code or the communities they lead

WHO CAN BE A GOOD ROLE MODEL FOR US?



- A person we can relate to (similar background, circumstances, characteristics...)
- Someone whose core values we respect
- Ideally: someone we can hang out with

MERITOCRACY & OPEN SOURCE

“

“Hackers should be judged by their hacking, not bogus criteria such as degrees, age, race, or position.”

- S. Levy
Hackers: Heroes of the Computer Revolution

E Mosido



ENGañADO

P

THE MYTH OF MERITOCRACY

- It does not exist neither in our society, nor in tech
- Read Toby Morris' *On a Plate* comic
<http://thewireless.co.nz/articles/the-pencilsword-on-a-plate>



THE PROBLEM WITH OPEN-SOURCE

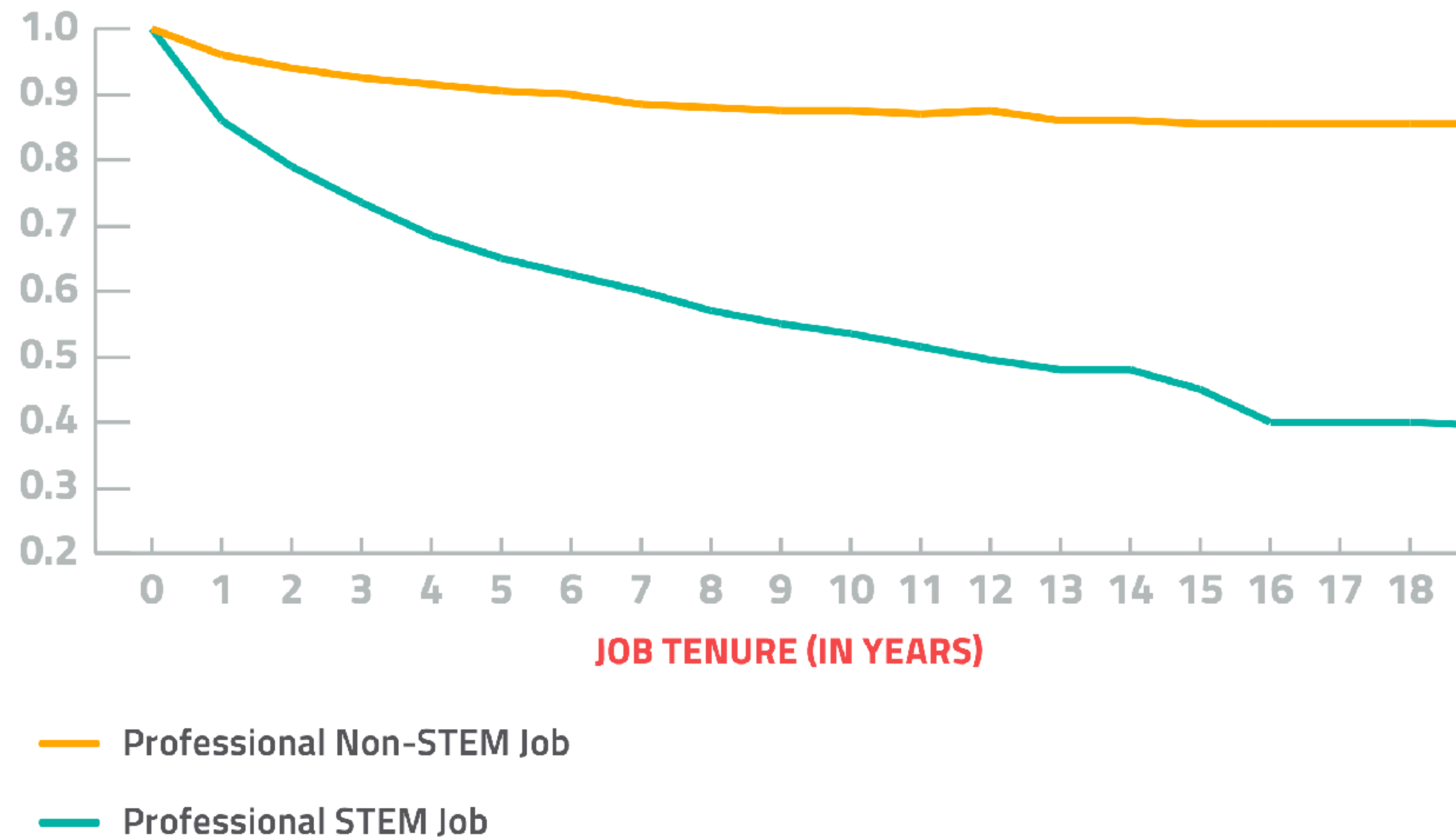


- On paper it's great...
- ...on practice, a lot of people get excluded
 - Lack of time and energy due to family care or housework
 - Lack of resources / space at home
 - A lot of people just can't afford to work for free
- Read Ashe Dryden's "The Ethics of Unpaid Labor and the OSS Community"

**WHY TALK
ABOUT THIS?**

**WE ARE LOSING A LOT OF PEOPLE
AND WE NEED THEM**

FIG. 1.6 // Percentage of Women Retained in Career Field Over Time




Source: Women in tech: the facts (2016 update)

IMPROVING OURSELVES

**IF WE BURN OUT, WE'LL
STOP DEVELOPING**

INCREASING OUR PERFORMANCE



- **Sleep** affects our learning, our performance, our decision-making, our stress levels, our *health*, etc¹.
- **Exercise** is crucial for health, energy and stress management
- **Meditation** to handle stress and improve focus (apps: Headspace, Calm)
- Try to be happy and fulfilled outside of work
 - Other activities can make us better devs (creative hobbies, sports, etc.)

1. Watch "What happens to your body and brain if you don't get sleep"


LEARNING

LEARNING: WITHIN THE COMPANY



- Both the company and the employee benefit from training.
- It's easier to get your company to provide you training for things that are related to your current job.
 - i.e. "React" training if you are a front-end dev, or attending Google I/O if you develop for Android.
- **There's a lot to learn in a 40h work week.** Learn from your colleagues. Read their code. Try new things. Do your best.

LEARNING: ON YOUR FREE TIME



- Can you squeeze some time from your day? (read on your commute, don't watch TV...)
- There are times when you just won't have the time... and that's OK.
- Sacrificing things that are important to you is not sustainable in the long term... so reserve this for special moments:
 - Landing your first job
 - Switching sectors
 - Learn a critical skill for a promotion you want

LEARN EFFECTIVELY

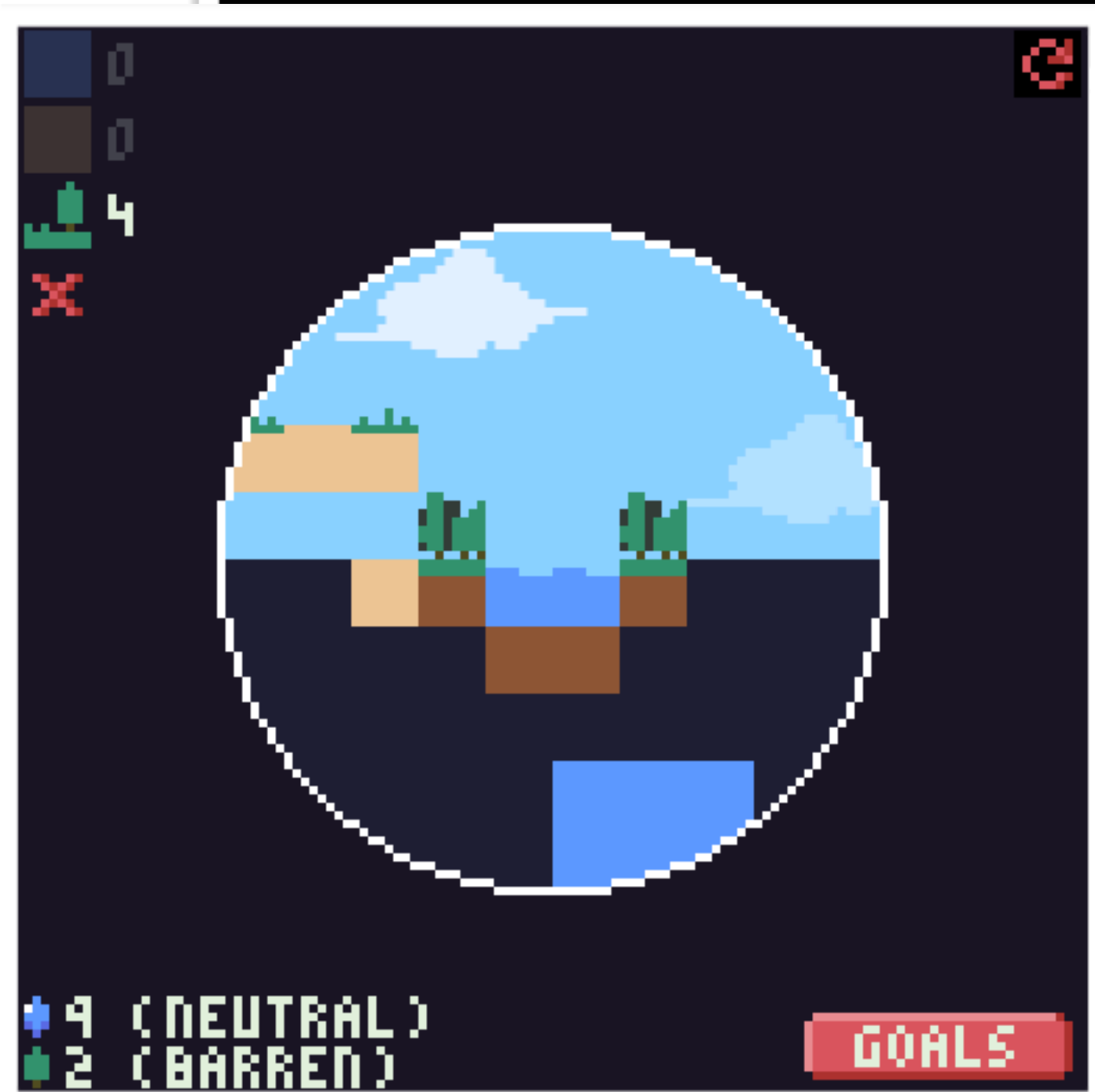
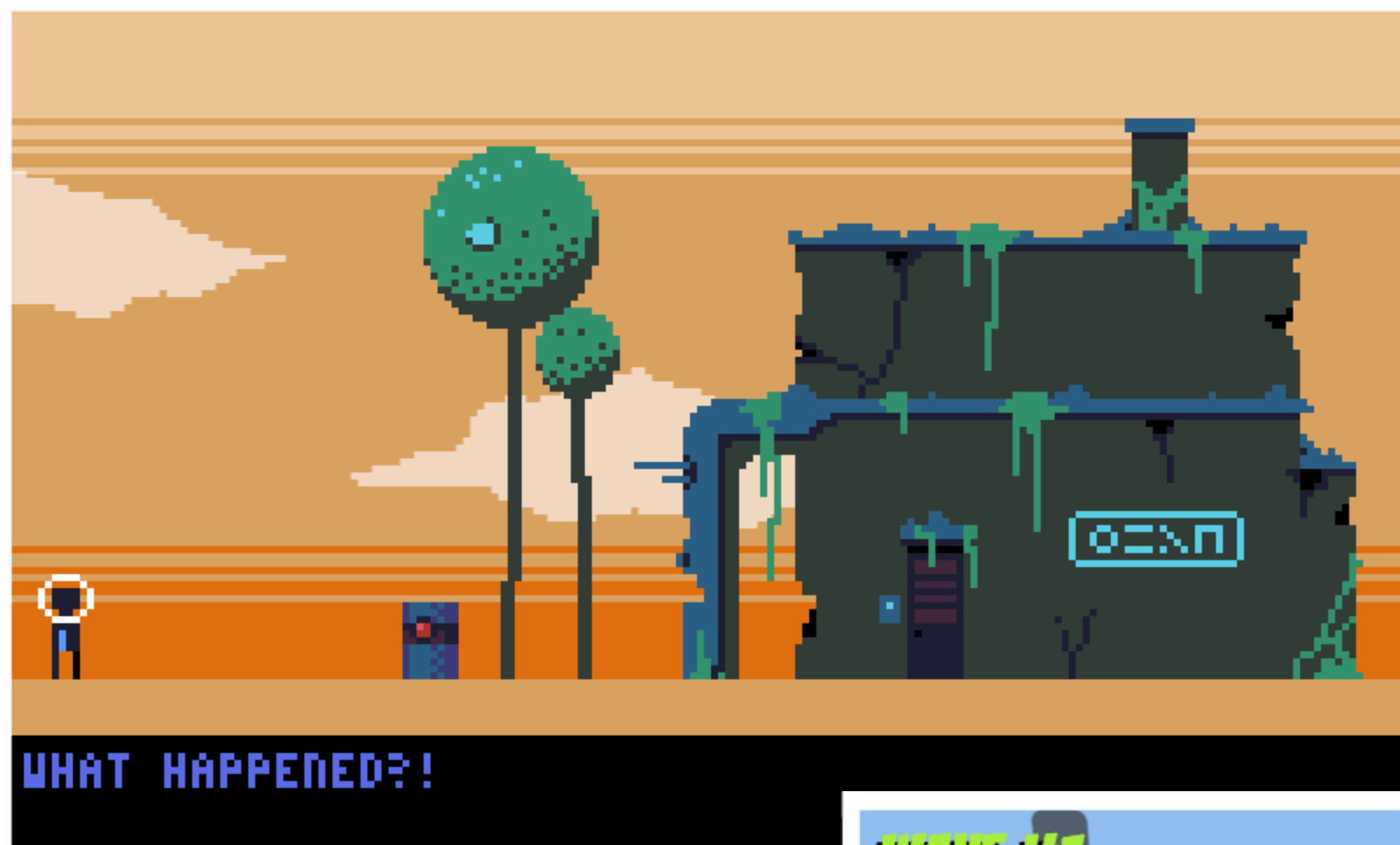


- Maximise your learning hours. Have a learning plan and don't go blindly.
- Enrol in a class / workshop.
- **Ask an expert** for resource recommendations, which topics to learn and in which order.
- You can repeat the plan for similar tech (i.e. create the same video game when learning different game engines).
- If you can afford it, don't be afraid of spending money on this.
- Teach what you learn to consolidate it.

SIDE PROJECTS & MOTIVATION



- Side projects are wonderful for learning new tech.
- ...but they are much more useful, career-wise, if we **ship** them.
- Ways to keep your motivation high enough to finish it:
 - Mini-projects (ex: hackathons or game jams)
 - Break a long project in small milestones... than can be shipped on their own
 - Dog fooding



The Packing Checklist

Pack this

Don't forget this FFS

- ☐ Medication
- ☐ Passport / ID card
- ☐ Health insurance card
- ☐ Tickets
- ☐ Wallet
- ☐ Credit card ⓘ
- ☐ Home keys
- ☐ Cash
- +

Do this

Weeks before

- ☐ Get visa ⓘ
- ☐ Get medical insurance ⓘ
- ☐ Get travel insurance
- ☐ Check documents are up to date

+

The day before

- ☐ Take garbage out

WHAT TO LEARN?

TRANSVERSAL SKILLS



- English!
- Abstract thinking
- Time management
- Focus
- Public speaking
- Reading & writing
- Professionalism

“EVERGREEN” TECH & TOOLS



- Version control
- Text editor (vim key bindings ftw)
- UNIX tools: pipes, cat, grep, sed...
- Scripts and automation
- Debugging tools

“ETERNAL” CONCEPTS



- Regular expressions
- Programming paradigms: functional programming, OOP, pattern matching, polymorphism, etc.
- Concepts / methodologies: SOLID, design patterns, data structures, algorithms, etc.
- Concurrency

IMPROVING OTHERS

MAKE YOUR TEAMMATES BETTER



- Your contribution to the team is not just your code... it's also how *you* affect other's code as well.
- If you are a senior developer, **this is your job too.**
- Provide constructive feedback (be extra nice, some people have trouble separating their ego from their code).
- Answer questions and be nice – don't make your team fear or hate you.
- Learn from each other: pair programming, code reviews, discuss together how to approach a problem, etc.

PSYCHOLOGICAL SAFETY



- Read "High-performing teams need psychological safety" article
- Study at Google on team performance saw that **psychological safety** was the common factor of their highest performing teams.
- In psychological safe teams, people are able to share ideas and execute them without fear of negative consequences, and they feel respected and accepted.
- Don't be the one who makes your team unsafe

YOU ARE NOT YOUR CODE



- And others are not their code.
- Don't diminish yourself (or others) because of a mistake. What is important is learning from it.
- Code is ideas, and we should be able to discuss ideas without personal attacks.

“SOFT SKILLS” ARE HARD!



- So called “soft skills” are usually dismissed because we are bad at them or we don’t like them
- They are crucial for your career
- Learn to communicate, to speak, to give feedback, to lead, to handle conflicts, to be a team player, to be empathic...
- ... and you can keep them from job to job :)

THANKS!