What I learned, notes - Week 1 Reset Phase

Getting to know the hub

- The Hub: A reliable repository of Nearsoft's internal content
- Why for?
    - Find a way to communicate the companies' internal content knowledge. Communication between the employees of the company it is very important.
- What can it do?
    - It has plenty information, where anybody can add a topic, which is suggested to always have updated. As well, you can learn from any company related topic

Lts & Wellbeing

The company has a flat organization formed by teams. The structure functions this way because you give empowerment to people, and as well you encourage leadership, self-organization and freedom. This also creates leadership teams (LT) to propose ideas and give follow up.

For Wellbeing a idea was proposed, because of this few things arose, i.e. gyms, physical activities, activities for mental health and nutrition.

The art of feedback

Feedback is very important to become part of the Nearsoft culture. There is no vertical growth as this encourage self-thinking and egocentric competition, because of this exist integral growth, which is multi directional and takes into account your working partner.

There are two types of feedback, the one that is given by the good and the other one by the bad. Even though is tough to receive or give "bad" feedback, it is necessary if you want to grow as person, but you need to be emotional aware if you want to give it.

The intention and context behind the feedback ends up being very important. Don't try to avoid feedback, it is inevitable if you want to grow, and most of the times don't take in a negative way.

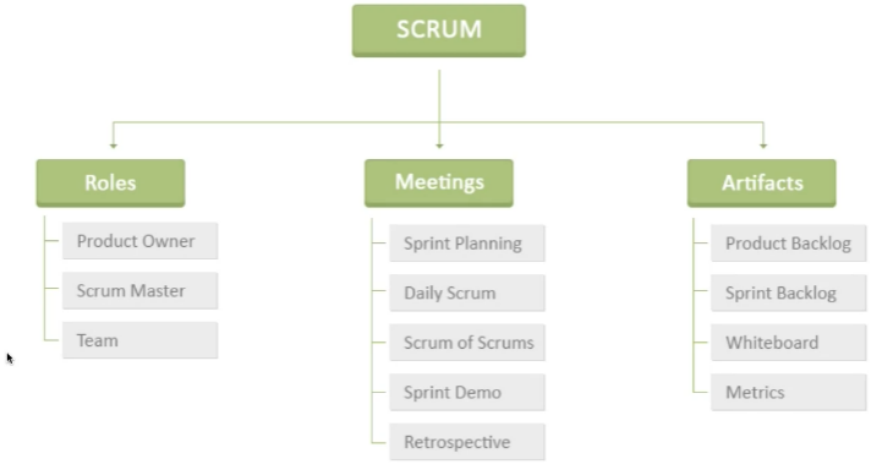
OMs

Atencion a la oficina, comida, eventos para las festividades, para los empleados, comidas esopntaneas, checan todo los de los viajes.

Agile class

Methodologies that do things in small increments/iterations. Some examples are:

- Scrum:



- Kanban:
- Scrumban:
    - Is a mix between both.
    - Do check ins of part of your code, not everything at once.

INVEST on Stories

Well-formed stories will meet the criteria of Bill Wake's INVEST acronym:

Independent	We want to be able to develop in any sequence.
Negotiable	Avoid too much detail; keep them flexible so the team can adjust how much of the story to implement.
Valuable	Users or customers get some value from the story.
Estimatable	The team must be able to use them for planning.
Small	Large stories are harder to estimate and plan. By the time of iteration planning, the story should be able to be designed, coded, and tested within the iteration.
Testable	Document acceptance criteria, or the definition of done for the story, which lead to test cases.

Stanford Lessons

- Programs should care more in simplicity, maintaining quality and managing complexity. Worry about performance at the end, when you have finished, maybe you won't ever have to worry about this anyways.
- Always prove what your intuition tells you

- When a new concept is introduced to someone, it must be followed by a vast number of examples so it can be clear and understand by the student.
- If your code worked well after you change something and fix the problem that you were trying to solve, then then you haven't fixed it. Use test cases to specifically identify which lines are the ones causing the problem.
- One idea of Agile coding, is getting a workable version of your code so you can start to test it, and start to solve problems that could pop even after you finish completely as no program is completely done after you think you have finished.
- Don't make things up, is better to not know than pretend to know.
- Coherence is unstable, but incoherency is stable as it has diversity.

Ideas are just a multiplier of execution

It is so true that you can have the brightest ideas, but as long as you don't execute them, they will stay like that, just like ideas.

Commands:

- Echo: print command
- Date
- Path
- Dir: show what is in that folder
- Cd: change director. cd . current folder, cd .. go back a folder
- MV: renames a file or move a file to another folder
- Cp: copy a file
- < file> file.ex : redirects input
    - missing:~$ echo hello > hello.txt
    - missing:~$ cat hello.txt:::: cat prints a file
    - hello

Network protocols

A network protocol is a set of rules which make possible communication between devices.

For us to understand network protocols and what they do, I am going to explain how this communication process works in the internet.

There are two types of models from where you can exapain how it works, the first one is the Open System Interconnection (OSI) Model and the other one is the TCP/IP model.

Following the OSI Model, we can divide by 7 layers

- **Application**: Human-computer interaction i.e. FTP (file transfer protocol), web surfing (HTTP/S), Simple Mail Transfer Protocol (SMTP), Domain Name System (DNS) How computers convert human-readable domain names
- **Presentation**: Data gets translated, compressed, and encrypted or decrypted.
- **Session**: Maintains connections and controls ports and sessions Verifies authentication and authorization, also manages files you have shared
- **Transport**: Divides data into smaller parts, manages flow control i.e. process velocity and error control (transport control protocol TCP, user datagram protocol UDP)
    - TCP: ensures data is received and knows when data is lost i.e. email, world web

    - UDP: Fast data transfer but doesn't care if data is lost i.e. streaming music or videos, gaming
- **Network**: Decides the path which the data will take (routing), ensures correct destination (internet protocol IPv4, IPv6)
- **Datalink**: Data is framed, provides access to the sending media and controls data transmission
- **Physical**: Sends signals depending on the media.

There are advantages and disadvantage to the network protocols we have just seen

The positive side free and open, they were meant to be this way so anyone can access them and know how they work, because of this it is free to communicate through the internet, but because of this anyone can know how messages are being sent and it is vulnerable to attacks and anyone can steal data from you, this is why we have antivirus and firewalls.

If it was private, you would have to pay to a company to use or buy their equipment, an example of this would be Skype

There a ton of Network Protocols, we mentioned the most important ones, they give an order to how communication must be done between devices

Pair programming

HTTP status codes

Git

Parents can talk to children but not the other way

How to talk to anyone

- People don't care how much you know until they know how much you care...about them.
- Don't diget makes people anxious
- Small talk is about putting people at ease. You must first match your listener's mood.
- By convincing them they are OK and that the two of you are similar.
- To talk somebody you want, ther eis no better way to just do it, maybe with a fact or recommendation or just, "I couldn`t help but…."
- Make yourself interesting
- Make them talk about themselves, people love it, when in doubt bring a subject you know they like
- To run from a conversation just make them repeat stories
- "How do you spend most of your time?
- Tell people about yourself but what is useful to them
- Wait for an appropriate moment to say "me too"
- Don't just smile like crasy to everyone
- Don't let alone a "thank you" always add something
- Read about everything, that way you will sound like a person who knows about topicsx that other people know
- Before every big purchase, find several vendors -a few to learn from and one to buy from
- Use their language so they feel you have a connection and to keep conversation clear
- Don't just hum or nod, empathize more, say more lebaorate sentences than just yep

- When agreeing, use words like I feel, I see, I hear.
- You can build history form the first moments of conversation with somebody
- Compliments from strangers mean more
- Whenever you hear something complimentary about someone, fly to them with the compliment.
- **Show appreciation**
- Telling him you admire him for the same reason he admires himself has an impact on Joe like no other compliment in the world.
- Even through the phone you transmit feelings. Be friends with who answers the phone if you call frequently. When hearing something in the background Ask whether she has to attend to it.
- Politicians always eat before they come to the party.
- People hate to be reminded of the moments when they're not shining.
- Only after they have played out this crucial charade can they discuss business. But no dirty business. The biggies can brainstorm over coffee. They can discuss proposals over dessert. They can toss around new ideas over cordials. They can explore the positive side of the merger, the acquisition, or the partnership while waiting for the check.

X Workflow

The whole company is a team. So learn of every group so you know how everything works.

When you start working on a problem, identify at what stage it is and follow it till the end. During the process you will need knowledge or tools that you don't know much about it, but **you will learn through practice it is ok not to know,** just don't take shortcuts or skip just because it is difficult.

Have a journal as ir will guide you if you ever go back or your peers want to consult your work. Somethings like:

- Initial Observations
- Initial Brainstorming Ideas
- Initial Questions to ask
- Background Info on the topic
- Possible search strategies to explore the topic
- Record of sources of background info
- Refined research question that could be tested
- Plan to answer the question
- Materials to use
- Data to be collected
- Reflections on research progress
- Reflections on data meaning

A **problem** rises through **observation** and with observation you can truly identify the problem reason or core. Then **gather** data historically. Then **analyse** your data to prove ideas wrong or right, use models for this, but use all the data possible. Generate a **hypothesis** use a Taguchi diagram to show the causes. **Experiment** (experimental design)**,** test several ideas and detect errors you could have made. **Test** these will be the protocols for anyone if they want to replicate your work and establish them clearly. Most of the time more problems will arise after the steps above, so **define your problem** in one sentence but considering everything. **Definition** should be a single sentence that express the problem, it is ok to have more definitions, but these are taken through what you

have already tested. Define **Constraints** as no single system behaves the same way. **Search for solutions** brainstorm, talk, group and gather ideas and test them against your constraints. **Research** as many problems have been solved already, so maybe your is too, but not completely in the same way, use a mix of solutions and their best parts to create your own. Write a **Scientific paper** as results must be share and shown. **Brainstorm** now that you have more information, go wild. **Model** as this will discard plenty ideas, and discard everything that doesn't work with the problem definition. **Prototype** because it proves a solution can be built. Create the **Specification** this is a guide with which you build the solution, a result of all previous work.

Now **choose a Problem** from the problems as your solution is not universal. Build a **Background** with al the previous work done in the problem by you or anybody else, as it is your justification, add your **Constraints**, then your **Hypothesis** but make it simple. **Propose a solution** just the general idea (use a "Black box" diagram). **Theory of operation** is just an explanation of how your program works (What does the user need to do to use it, what is the sequence of execution, how the data flows, how does the user receive the result of the program.). Next the **Black box,** what does the system receives, a simplified version of the system and what does the system produces. Add a **Flow diagram** which is the sequence and steps of the program. **Functional Specification** the detailed version of the solution proposal (use diagrams, storyboards and text), describe all the components. **Technical specification** describes the steps to implement the solution with color codes, algorithms, etc, anyone following these can implement your solution, sort of like a patent. **Implement** everything, begin execution. **Data analysis,** check what you just did, and check if it is right, review you constrains and hypothesis. Finish by **Documenting your work** and write about improvements, new ideas, if you failed or succeeded.

Jonathan Haidt Religion, evolution, and the ecstasy of self-transcendence

Everyone has their own mode to self transcending. War strongly brings people together, the "I" turns into "we, the self thins out and goes away.

When Individuals unite into a community uplift. Collective feelings are stronger.

Our virtues shine better in groups than alone.

From one group cooperating between each other sometimes one "free rider" appears, if this is not controlling the free rider will contaminate everybody else selfishly, and there will be two groups separately. The best way to solve this is join the two groups and grab the better things of both and create a "supergroup".

Pursing communal interest is stronger than a self-drive. People are meant to cooperate with other and purse ideas. Modern society has guide us to have freedom, and is good, but we only desire mundane things and not what we could really strive for.

How progress really happens

You must want to make change.

Some people are more open to change than others

All types change demand some work. People resist change because it requires work, is difficult.

Why do people who live in flood areas don't ever change? Because they resist it, traditions fight change.

Idea killers:

- That's not how things are done
- We have already tried it

These killers help you to solidify your idea. Change comes when things are unpleasant, if you want to make change find someone who is unhappy or unsettle. Also, power makes change happen.

Use "grass roots" are ideas or initiatives a that potentially will create change.

Intuition helps you to identify people who are open to change and anticipate how reactions will come out.

If you have a really big idea, narrow it down into a "sub-idea" to start with and attract people to change, it is better to start small and keep piling up and start growing and getting support.

People make change not tools!

Creative thinking

Creativity is a kind of work

All ideas are made of other ideas, if you don't know what to do, just combine ideas.

Everything is recycled to create new things, these are inspirations for you.

Creative thinking hacks:

- Combination
- Inhibition
- Environment
- Persistence
- Hacks

If you identify constraints, assumptions or fears and remove them you become more creative.

"I want to be different, like that guy", it is your uniqueness that makes you creative and different. Walking or moving makes the brain more comfortable when thinking, as well as touching or feeling things. Writing sometimes is better than typing for getting ideas, you think differently.

Idea generation. Work hard and work smart, persistence is good but most of the time you don't have to be.

Hacks:

- Journal (memory is awful) record your ideas
- Escape being alone
- Invert solve the opposite problem from the one you are trying to solve. No wrong answers for an idea, freedom to brainstorm.
- Partnering, ideas come from different minds. If there is no partner find a rival.
- Fail, take enough risks that you fail, if you aren't failing you aren't being enough creative
- Switch modes, some ideas are easier to discover through: visually, verbally, physically audible. Find a different way to represent a problem.

First draft of anything is going to suck

Don't bring labels into things, so don't go and just say be creative

How to master anything

Not all the time practices make master.

Deliberate practice -> mental representation (good quality and quantity)

Climbers know every move of the rout before climbing it.

Master have achieved mastery as they can recreate experience mentally, make accurate predictions. They have excellent mental representation.

How to achieve this?

Purposely practice:

- Have a specific Goal
- Intense focus for periods of time
- Immediate feedback
- Push constantly into discomfort zone

Force mental adaptation -> spark creative insight

Steve experiment (remembering numbers), he created a technique

After this the experiment it was done with another person but this person was fought Steve technique and there was more rapid improvement, but he thought that technique had flaws and develop his own technique.

Purposeful practice methods + expert couching = deliberate practice

For improving:

1. Get yourself into deliberate practice sessions daily
2. Discover new mental representations

Badass developers

Reduce the use of cognitive resources because these equal to willpower. Cognitive resource management is expertise. If things are difficult you burn cognitive resources. Don't burn yourself out, dog example.

The 3 steps of things are:

1. Things you can't
2. Things you can do but with effort
3. Things you've mastered

Don't try to learn an entire skill and go through every each one and master it. Is better to divide into sub skills and move from 2 ->3

Practices makes permanent

You could jump from 1 -> 3 but is that kind of skill that you learn and you "just know"

Chicks or plane example

This is called Perceptual learning

High quality

Very high quantity Examples

This will create better pattern skills and you will learn faster