Clean Code - Episode 1 - Why does code rot?

- Code rot - Software rot, also known as bit rot, code rot, software erosion, software decay, or software entropy is either a slow deterioration of software quality over time or its diminishing responsiveness that will eventually lead to software becoming faulty, unusable, or in need of upgrade.
- Greenfield Project - A greenfield project is one that lacks constraints imposed by prior work on the site. Typically, what a greenfield project entails is development on a completely vacant site. Architects start completely from scratch.
- Brownfield Project - A brownfield project is one that carries constraints related to the current state of the site. In other words, the site might be contaminated or have existing structures that architects have to tear down or modify in some way before the project can move forward.
- Requirements Document - A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders (business, users) needs.
- The Big Redesign In The Sky - Many software developers take this to mean that if you have a huge legacy mess in your software you should stop working on it and rewrite it from the ground up. A race between new and old systems.
- Rigid System - changes are difficult to make and therefore, estimates aren't accurate.
- Fragile System - software brittleness is the increased difficulty in fixing older software that may appear reliable, but actually fails badly when presented with unusual data or altered in a seemingly minor way
- Inseparable System - where you can't reuse parts/modules in a system for another system. Unpredictable, because you don't know if you can reuse the code, so changes cannot be reliably estimated.
- Opacity - Tendency of a system to be so ineptly structured that you can't understand the author's original intent. Reading the code tells us nothing about what the system does or the way it works.
- Clean now, not later - The only way to code fast is to code clean - deadline pressure and rushed code will only make things worse in the long term - clean the code there and then.
- Bjarne Stroustrup (inventor of C++) said: 'I like my code to be elegant and efficient. Clean code should do one thing'.
  - Elegant code - code that does a lot in a few words.
  - Efficient code - code that uses very few CPU cycles.
- Grady Booch (Object-Oriented Analysis and Design): 'Clean code is simple and direct. Clean code reads like well-written prose'.
- Michael Features (Working Effectively with Legacy Code): 'Clean code always looks like it was written by someone who cares'.
- Ward Cunningham (inventor of wikis): 'You know you are reading clean code when every routine that you read is pretty much what you expected'.
- The Boy Scout Rule - 'Leave the world better than you found it'