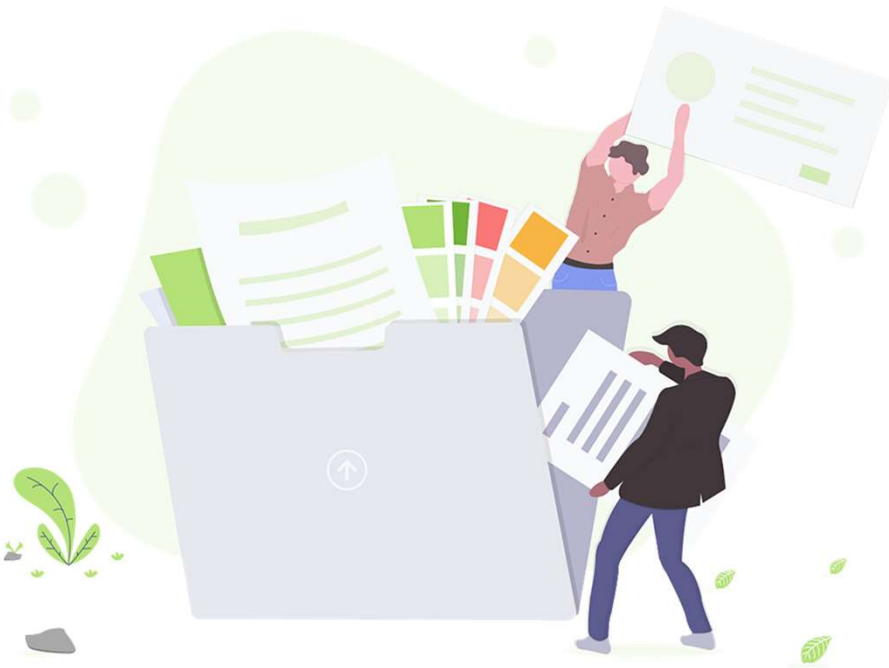
Top-Down Approach in Programming

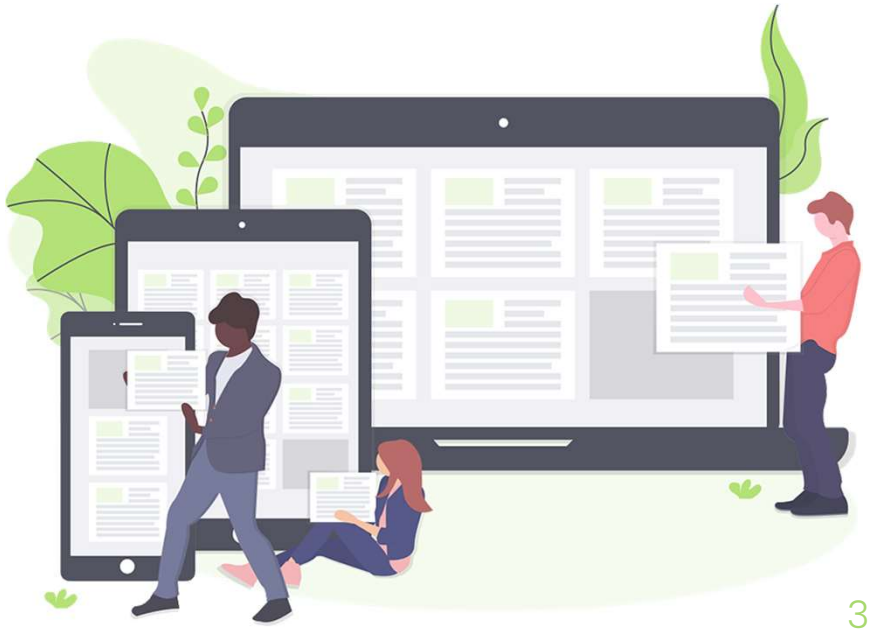
Name: Le Van Ky Du
Date : 13/01/2024

1. Introduction



What is Top-Down Approach in Programming?

- With this approach, the main goal will be to learn by creating (small pieces of) real software.
- Most self-learners will follow this approach when they first start learning to code. Because they simply want to be able to create attractive things immediately like a website or a 2D, 3D game...
- And the fastest way to do that is to learn to follow a tutorial - long and specific, a tutorial with very detailed instructions. If you follow the steps correctly, you are guaranteed to create something (interesting to you).



Top-Down: The Good ?

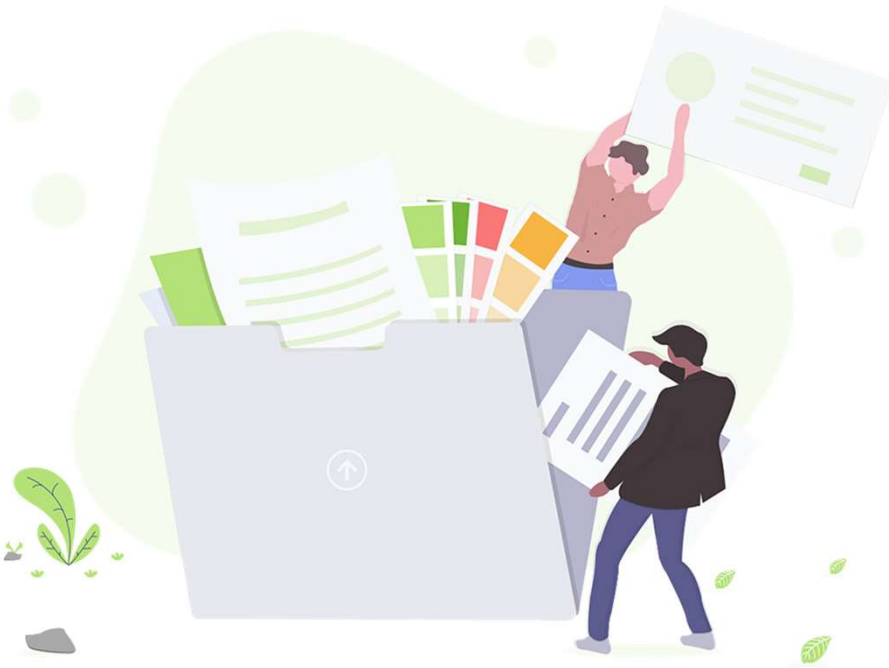
- "Organization”: Breaks complexity into manageable steps.
- "Efficiency”: Flexible and easy to debug.
- "Understanding”: Clear and maintainable code.
- The nice thing about this approach is that you create something very quickly. It feels great to create real software with your own hands in a short time. You can get addicted to that feeling , and it can help you stay motivated through the studying process.

Top-Down: The Bad ?

The problem with this approach is that it doesn't teach you basic definitions or principles.

At the end of each tutorial, you may not understand how your product (software) works. Even if a tutorial is not detailed enough, you may struggle to solve its problem; when you get off track, even the slightest, you may not be able to complete it and be unable to diagnose it. problem. It's like being thrown straight into the deepest part of a swimming pool before you even know how to swim.

2. Applying the Top-Down Approach Process



To applying the Top-Down Approach Process

1. **Understand the Problem:**
   - "Define the problem to be solved."
   - "Identify the major components."

.

2. **Design main Module :**
   - "Design the highest-level module of the program."
   - "Define the main functions."

To applying the Top-Down Approach Process

3. Identify Submodules:

- "Break down the main module into smaller submodules."
- "Each submodule represents a specific task."

4. Design Submodules:

- "Further break down functionality into smaller steps."
- "Design interfaces for each submodule, specifying inputs and outputs."

Thanks!

Any questions?