Creating Developers

Khwarizmi Project

This framework powered by

Khatwa initiative, Coders Making and Others.

Created in

18 Jan, 2020

V. 1.5.0

Contact us: Khwarizmiidea@gmail.com

Introduction

Software Development?

The main reason for software Development everywhere is the communities, for instance, markets and investments depend on communities, thereby the community should be aware of the core value of software development. In Sudan, the matter is that communities are ignorant of the significance of software which causes many other problems ,such as reducing the salaries of software engineers ,etc. and this causes a scarcity of programmers.

How to tackle this problem ?

To tackle this problem we need to build a new community and be aware of individuals among this community but this requires a lot of effort and plans ,that may succeed or fail.

Khwarizmi project?

Khwarizmi project is an open source project.which Many of the initiatives and programmers discussed this problem and the outcomes of that debate was, and the name of Khwarizmi was chosen relative to the great Arab scientist Muhammad bin Musa al-Khwarizm.

Developers tried to make a solid plan to build a good software community in Sudan. They wanted to apply some rules or methodologies to aware people in the software industry, after a lot of effort they came up with this project.

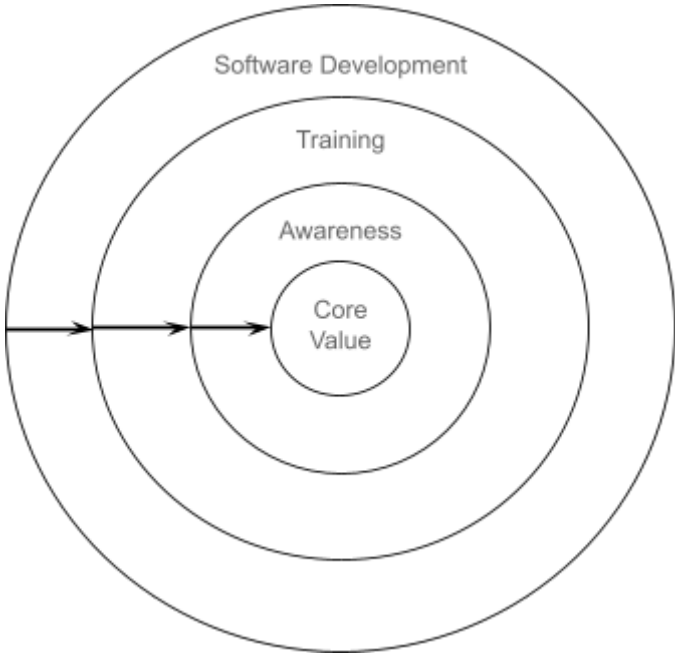
Khwarizmi project tackled this problem through?

creating a lightweight roadmap for people to learn how to create software perfectly, this process could be divided into following four main parts:

1. Core value.
2. Awareness.
3. Training.
4. Software Development

The Key Point of This Project:

To have a progress and knowledgeable community in software .we have come up with these following main elements that support software development which integrated into below diagram:



note: the arrow on the left side indicates that each outer layer depends on its inner layer, for example: in order to be a software developer you'll need to have training, and so on and so forth.

1. Core Value

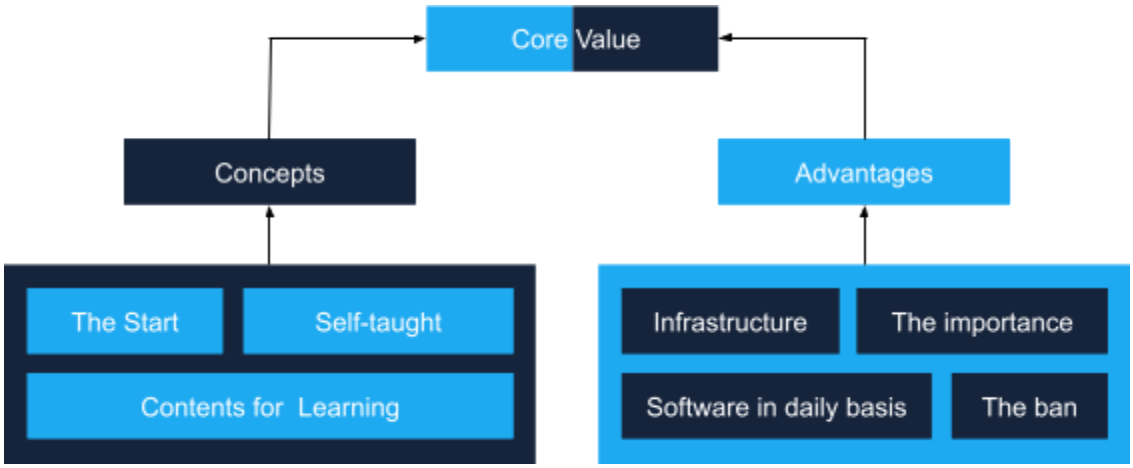
The core value of software development in general is so important because software development is taking place today, everything now can be made with software or translated into a software, so it's worth a lot to invest time and money on designing or developing software projects.

2. Awareness

There are many people who started learning software development, but lots of them end up frustrating, they will likely give up, and that because they started in a wrong way or maybe they picked up a wrong path, or the tool they use is not appropriate or inconvenient with them. But if we follow some strategy to not let people give up, then we are applying to Khawarizmi. The first before we start teaching people the tools/languages, then next points should make sense to them.

1. The value.
2. Advantages.
3. Software on our daily basis.
4. Needs for software.
5. How to start.
6. Contents for learning.
7. Self-taught.
8. Concepts.
9. Infrastructure.
10. Electronic ban in Sudan.

We tried to order the ten points above by their priorities in the next flow:



3. Training

Training is the third part of this project, and it would be valuable if people recognize the whole value of being a software developer. In order to teach people to build and/or develop a software we must apply some effective rules on the main platforms (Desktop apps, Web apps and Mobile apps) and the rules are:

1. UI Design.

   User Interfaces are parts which are used by clients, the UI is the actual interaction process that occurs between clients and software. The nicer and easier user interfaces, the faster your software gets widely shared. So in the training part people should get the full benefit of designing fast, easy to use, rich and fully interactive user interfaces.

2. Logics (e.g. handle data, validation).

   The thing that defines the software behavior is the logic that happens behind the scenes, like validations when you want to check whether the data in the user form is valid or not, and if not for example you throw an error and the user will directly get a kick back with what happend. Every software has its own behavior and logic depends on what software is made for.

3. Database Design.

   The database is essential in any software, and there are a lot of ways to implement a database, which is based on usage of software itself. Database design is so important, so people should definitely know the main concept of database and the right way of designing well-structured databases

4. Design Patterns.

   In this stage developers should learn the benefits of well-structured code and also several ways to design the code. There are many things that would take place in this stage such as a highly designed Separated decoupled system.

5. Codetesting.

   Code testing is the most important phase in the lifecycle of software development. Most of the developers don't actually test their software to know whether it functions well or not.

In this project we will teach developers how to write a code that tests the code (Testing), in the software they create.

6. Deployment.

   Finally we want to deploy the software to the world after finishing developing it. Here the developers will know everything about deployment, and they will use several different tools in this phase.

7. Maintenance.

   After the deployment we might have to take time maintaining the software we made and fix some bugs if there, so we have to make sure that we professionally developed it.

8. Evolution.

   Evolution of software is so important. The software itself might be good, robust, fully-featured and easy to debug, but if it's good that doesn't mean it won't change in the future, at least it's behaviors, so the evolution of any software will always exist.

Training is project-based.

In the training part, people should be taught in projects, this architecture will improve developer's skills and let them get used to the environment of software development as a team work or individual.

Training tools /Software languages?

Choosing the tools for training should be based on the market and community for online/offline support process, if there is an issue.

Criterias for selecting tools / programming languages.

1. Labor market.
2. The ban.
3. Open source / closed source.
4. Free / paid.

5. Supportive community.

Note for training.

How to think as a real developer and being a problem solver. How to create teams and picking the right people to be part of your team, and the most common goals to achieve as a team.

Problems that encounter the training phase:

1. Transportation.

   The proposal solutions for this problem are:having a means of transportation as a car or renting a car , Sharing transportation cost, Online Training.

2. Train the trainer before bootcamps.
3. Time management.
4. Trainer's experience.

   Trainer's experience includes: Previous projects, Contents and mentors.

4. Software Development

The most important thing that the software developer needs in developing software is that ethics you might not be able to create the most complex ,useful software without having ethics & good manner .this issue may depend on awareness and training rather than being taught in the development stage:

There are some key-points we should care of:

1. Soft-skills
2. Experience
3. Volunteer
4. Business