Classification of developers by martial arts ranks [Junior, Middle, Senior]

In many martial arts, the color of the belt corresponds to the skill level. Usually, the color changes from white to black, where the experience level corresponds to the darkness of the belt. A beginner wears a white belt because he has no experience. White means “new and clean”. As you train, the belt darkens, showing progress. The color represents the dirt accumulated through hard work and sweat. A martial artist with many years of experience eventually reaches the black belt, which means a high level of knowledge and skill. Traditionally, the belts were only black or white. But in recent decades, more colors have emerged. The same in our company our developers are going through a challenging trip to gain the “black belt”.

How do junior, middle, and senior developers differ? How does the transition between these levels take place? For these and many other questions, our experts provided detailed answers.

Basic Characteristics

For us, “Junior”, “Middle” and “Senior” levels mean not only technical skills (hard skills), but also the soft skills of being able to communicate with people, working in a team, and the willingness to take responsibility for own decisions.

The higher the developer is positioned; the higher requirements for soft skills are. Technical skills are usually the easiest for us and developers to develop in comparison to communication and teamwork skills.

Experience is a rather conditional characteristic. Some developers can be classified as newcomers even after 4 years of work in the company, but there are real “diamonds” who, just in a few months after entering the market, already show independence and do not continuously support their senior colleagues.

An important element in evaluating the developers’ experience depends on their background performance and future potential. Sometimes an engineer who works for a small company that is engaged in streaming development of similar projects can hold the position of a senior developer in that company, but at the same time, if he jumps into an ocean of a large high-tech firm, his level can be assessed lower or higher. Thus, we get a person who is coming with a background from one company, and he still might need to go a long way with us increasing her/his professional level.

World statistics say that a junior developer is defined as a beginner with real work experience of up to 1-1.5 years, a middle programmer as a still learning specialist with 1-3 years of experience,

and a senior as a professional who has worked in the company for a good 5-6 years. But remember that this gradation is an approximation. Therefore, the level of the developer should be determined depending on the **professional skills, knowledge, and characteristics** as following:

1. Computational technologies (Computer Science): data structures, algorithms, system programming;
2. Software Engineering: VCS, IDE, CASE, CI / CD, middleware, processes, metrics;
3. Programming: programming languages, libraries, frameworks, code organization, organization of own work;
4. Communication skills;
5. Cognitive skills;
6. Knowledge of the subject area;
7. Experience.

***The first two disciplines are often overlooked, but they play a large role in building an effective development process and have a significant impact on the quality of applications developed. Sometimes we meet developers who apply for Senior developer, while do not know what a stack or a graph is. The question arises if a person does not know what a stack is, then how affair example, a debugger?

If a person does not know the development process, then how can he effectively part know addition to other obvious statistics and knowledge, I would like to draw attention to **cognitive skills**: the profession of a developer requires constant training not only to grow but also to stay at your level, keeping up with the times. And here memory, attention, the ability to focus, and the speed of information processing play an important role.

Definition and Differentiation

The specific content of the levels depends on the technology stack that is used in the company.

Junior developer: white belt

is usually a person with little or no development experience. In our cases, s/he’s a yesterday’s student or even a student who still studying at university with a random set of initial skills that we thought was sufficient to give the person a chance. S/he is ready to listen to criticism and learn a lot.

You need to understand that for tasks that the signor will solve in ten minutes, Jun may need three approaches an hour each, and in the process the code will have to be completely rewritten, spending a lot of additional energy.

Middle developer: blue belt

is yesterday's Junior, who has successfully mastered the entire technology stack used by the team. He confidently, independently and on time solves small problems/bugs. Provides useful remarks when reviewing someone else's code. In addition, he knows several programming languages/frameworks, and for what his/her knowledge is systematized. Additionally, the middle developer can independently evaluate his part of the project and start developing it without additional assistance.

Senior developer: black belt

is mentor, evangelist. Due to his/her deep understanding of the system architecture, s/he can be entrusted with a new product or direction. This expert already runs a team (team lead) or is a very cool developer (tech lead). S/he understands for whom this or that product is being made. Who should do what and how?

Moreover, the Senior developer sees the picture of development as a whole, presents the complete architecture of the project, and understands what should ultimately come out in the release part.

There is also division within these concepts. Besides Junior, there is entry-level Junior, med-level Junior, and plus-level Junior. Likewise with Middle and Senior. We focus on these levels when looking for new programmers and working with those who are already on the team. Looking at the requirements, and they are fixed and open, employees understand in which direction to "dig" to grow. This is something like OKR (Objectives and Key Results).

If we go into more detail, when we have something to add to the above mentions:

What should a Junior be able to do?

- Possess the basics and programming tools;
- Speak basic English;
- Be able to write basic program code;
- Possess the skills of reading code;
- Be able to listen and hear criticism, improve his/her mistakes;
- Navigate the IDE interface and manage it using the taskbar;

- Know how to work with the API.

Skills required for a mid-level developer:

- Knowledge of keyboard shortcuts for quick and efficient work with IDE;
- Writing understandable code, confident knowledge of technology;
- Database management and development: provision of stored procedures, triggers, user-defined data types (UDT), knowledge of object-relational mapping (ORM) technology;
- Having a deep understanding of the functioning of more than 4 platforms;
- Active collaboration with team members, guidance over junior developers;
- Independence in work;
- Search for non-standard solutions;
- What are the responsibilities of a senior developer;
- Pre-intermediate or intermediate English.

What are the responsibilities of a senior developer?

- Technically consistent with the business product program;
- Have leadership qualities, be able to manage a team;
- Fully understand the architecture of the application;
- Constantly improve qualifications, learn new things;
- Be able to calculate an accurate estimate;
- Know how to independently implement a project from scratch;
- Solve any problems that arise;
- See the strategic path from an idea to its successful implementation and market launch;
- Intermediate or fluent English.

Higher-level moving

Junior → Middle

This way, they will fall into many possible pitfalls and learn how to avoid them. They learn how to write simple code by thinking of the person who will be working on the program after them. Also, learn how to fix bugs and educate themself.

At this phase, they learn how to debug, as this will help better understand what's going on in the process. In addition, they are familiarized with best practices and learn about architecture, performance, security, and more. Close the knowledge gap required to move to the middle tier. For a junior developer, learning design patterns, architecture, test automation, performance, and security techniques, and more is a great way to close the knowledge gap.

Middle → Senior

Moving from middle to senior level can be quite challenging, usual now the focus is on learning, not just on solving routine tasks. Some developers stay middle for their entire careers. Seniors know what can be discarded in the code, and what cannot be removed in any case. Previous experience and mistakes taught them all this. If a developer wants to be a senior, then he/she should be prepared to take on tasks that no one else can do. Also, to have to help less experienced developers, Seniors are their lifeline in difficult cases. It is not surprising that seniors thoroughly study the entire range of technologies of their company. This is more than just programming – it is an immersion in all aspects of creating a product.

Seniors need to know more than just how to get the job done.

What's next?

The Senior Developer position is precepted by us not as a career plateau, but as a springboard for further development, for example, in one of the following areas:

- Technical expert;
- Industrial expert;
- Front man;
- Team lead;
- Architect.

Serious fighters have been learning martial arts all their lives; serious software developers do the same. Any knowledge in this area is completely outdated after 3-5 years, and if there is no constant growth, they can "slide" from senior to the middle position again. Therefore, we recommend to all programmers to constantly educate themselves and keep their fingers on the pulse of technology development.

We recommend reading new articles and research on the topic, testing new products and technologies. This is what always keeps you afloat and makes you competitive.

We hope you find useful insights based on the differences between these three developers' levels.