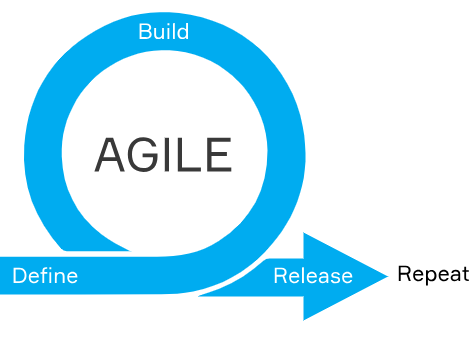
Imagine your development team has received an order for a project. Now you need to come to an agreement with the customer, distribute tasks among the team members, and do everything on time. In order to make a good project, it's useful to take into account not only the costs and possible errors but also the human factor. That's why Agile was invented.

Agile is a group of models that are guided by the same principles of interaction between developers and customers. It differs from the SDLC we talked about earlier. SDLC models are used in project management to execute the software development lifecycle process. Agile is a whole methodology for this process where the customer's requirements will be fulfilled by the means of collaborative work. In this topic we will list these principles, explain how Agile simplifies teamwork and what models are included in the Agile group.



**Agile principles**

All the principles of Agile are described in the Agile manifesto. Below are the main ones that developers should always keep in mind. So, during the development they need to:

* deliver the necessary software continuously and without interruption
* be open to product changes at any stage of development
* constantly be in touch with the client
* communicate in person
* stay motivated
* be well-organized
* try to implement technical improvements and user-friendly design

These principles are the main idea and philosophy of Agile. They are very important since taking into account the human factor and explicitly describing the tasks and goals of communication can significantly speed up work and increase the effectiveness of the result. However, this manifesto is not the only thing that is included in Agile. There are also technical concepts like **sprint**and **issue trackers**. We'll discuss them below.

**Agile components**

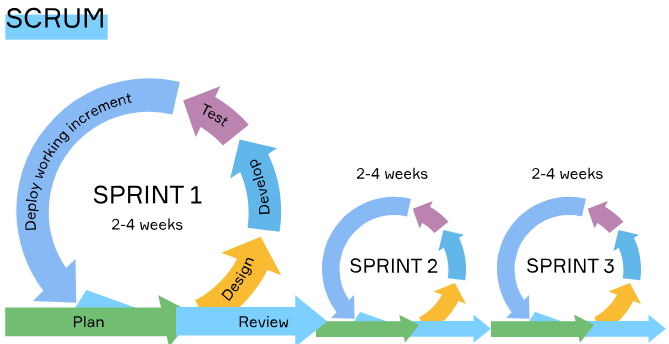
In the course of work, various bugs and shortcomings may arise that need to be taken into account, entered into the plan, and fixed. This can be done with the help of an **issue tracker**. An issue tracker is a software that manages and maintains lists of issues. Issue trackers are commonly used in collaborative environments, especially in large or distributed collaborative projects. These systems often include resource allocation, time tracking, priority management, and workflow oversight. More specifically, for each bug, a card with an identifier is created, which is called a **ticket**, so that the error is recorded and can be found by its identifier. It should also be noted whether each error was resolved or not. The cards are placed on the board and distributed among the team members. The most popular examples of such boards are **Jira**, **YouTrack**, **Bugzilla**, **Trello**. Consistent use of a tracking system is considered one of the "hallmarks of a good software development team."

The other important notion is a**sprint**. Agile modes are iterative and may have several iterations while in the development stage. Each iteration is called a sprint, i.e. some part of a project that needs to be done right now. Sprints do not take too much time. Usually, they last no more than 4 weeks depending on a specific agile model which we will discuss below.

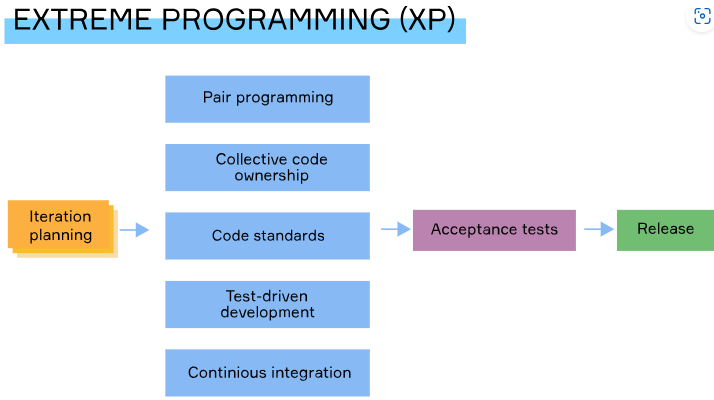
**Agile subtypes**

Agile comes in different flavors. Today, its most common subtypes are Scrum, Extreme Programming, and Kanban.

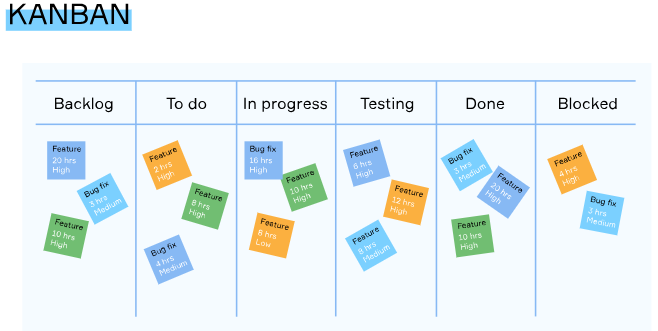
* **Scrum** is probably the most popular Agile model. Each sprint lasts from two to four weeks, during which the developers can make changes in the project and consult with the client. After the end of the sprint, they do not return to this step anymore.



* **Extreme Programming**. Its iterations normally last 1-2 weeks. All development stages are carried out simultaneously under extreme conditions. At any time the team is ready to make changes according to customers' requirements and immediately test them. Developers work fast and in pairs, like one writes code and another one checks it. This approach can be very effective, but it requires a lot of effort from the team. Therefore, it is better not to work in this mode for a long time. Plus, the team should not be very large, otherwise, it will be difficult to control and organize the teammates in such an extreme mode.



* In **Kanban** the emphasis is on visualizing the plan. Team members draw it up on a kanban board, another type of issue tracker we discussed above, where all tasks with their timing and responsible individuals are clearly spelled out. The work goes on every day strictly according to plan. Daily negotiations with the customer are possible, and there is an opportunity to make changes at any time. Due to its nature, the model is often used in software support and development projects.



Each of the Agile models we reviewed has its own pros and cons. The main thing is to agree in advance with the team and the customer which model suits you best and follow it. If you are confident that you can work in extreme mode, great, you will choose extreme programming to make edits on the fly. If you want to work in parts and then not come back to them, choose Scrum. If you like to visualize the plan, take small steps every day, the Kanban model is just for you.

**Conclusion**

So, we have analyzed Agile and the models that it includes. Agile principles are widely used in different companies and help to optimize the work of the team, improving the quality of the result. We hope that you understand now that everything is important when working on a project, every detail, and that you will not have any difficulties in working on Agile when you get a job in IT.

## Appropriate Agile model

Match types of Agile models to their typical use cases.

Match the items from left and right columns

Here is the correct matching:

* **Scrum** → *work in parts, make changes and then not come back to them*
* **Kanban** → *create an exact plan and work step by step every day*
* **Extreme Programming** → *work fast in extreme conditions when there is little time and a lot of tasks*

## SCRUM sprints

What is the average length for SCRUM sprints?

Select one option from the list

Two to four weeks

## KANBAN boards

Which of the following are KANBAN boards?

Hint: They mean: in which tools you CAN display kanban board. Because kanban is very widely used, every advanced tool can display it.

The correct options are:

* ✅ **Trello**
* ✅ **YouTrack**
* ✅ **Jira**

**Explanation:** These tools support Kanban boards for task and workflow visualization.

❌ **SCRUM** is an Agile methodology, not a software/tool.

## Agile main idea

What is the essence of Agile principles, what does it help us with?

Pick one correct answer.

Select one option from the list

Working in close collaboration both within the team and with clients

## Agile components

What other components does Agile include (besides the list of its principles)?

The correct components that Agile includes (besides its principles) are:

* ✅ **Sprint**
* ✅ **Kanban board**
* ✅ **Issue trackers**

### Explanation:

* **Sprint**: Core to Scrum, a type of Agile framework.
* **Kanban board**: Visual tool often used in Agile to manage workflow.
* **Issue trackers**: Used to log and track tasks, bugs, and features—common in Agile environments.

## Agile principles

The principles of the Agile can be found in the Agile...

Select one option from the list

Manifesto

## Agile subtypes

What separate subtypes of agile have we discussed in this topic?

Scrum

Kanban

Extreme Programming

**Extreme Programming**

What is included in Extreme programming? Choose several correct answers.

Select one or more options from the list

The correct components included in **Extreme Programming (XP)** are:

* ✅ **Iterations of 1–2 weeks**
* ✅ **Simultaneous development stages**
* ✅ **Pair programming**

### Explanation:

* **Iterations of 1–2 weeks**: XP emphasizes short, frequent release cycles.
* **Simultaneous development stages**: XP encourages overlapping phases like coding, testing, and integration.
* **Pair programming**: A hallmark of XP—two developers work together at one workstation for better quality and knowledge sharing.

**Kanban usage**

What is the most important part of the Kanban model? Pick one correct answer.

Visualization of the plan and strict adherence to it

**Scrum usage**

The development team chose the Scrum model and have already completed the first sprint. They showed the result to the customer, but he was not satisfied. Can the developers make changes to the already finished part of the project within the framework of the chosen model? Pick one correct answer.

No, they cannot make changes after finishing sprint.