



Jira Essentials with Agile Mindset

1

Course Overview



11 October 2023 – Cloud

What will you learn?

- Use lean and agile principles
- Differentiate kanban and scrum
- Configure Jira to match your team's current process



To succeed here, you need to have

- No knowledge of agile or Jira is assumed



Live teach schedule (1 of 2)



| | | |
|---|--------------------------------------|------------|
| 1 | Course Overview | 20 minutes |
| 2 | Agile and Jira Overview | 45 minutes |
| 3 | Visualize Work Using Project Boards | 35 minutes |
| 4 | Enrich issues | 40 minutes |
| 5 | Kanban Method | 30 minutes |
| 6 | Scrum Overview I – Artifacts | 15 minutes |
| 7 | Scrum Overview II – Roles and Events | 35 minutes |
| 8 | Quick Search and Basic Search | 25 minutes |



Live teach schedule (2 of 2)



| | | |
|----|---------------------------|------------------|
| 9 | Filters | 15 minutes |
| 10 | Epics | 20 minutes |
| 11 | Dashboards | 15 minutes |
| 12 | Putting it all Together | 15 to 40 minutes |
| 13 | Lean and Agile Principles | 45 minutes |
| | Total | ~7 hours |



Zoom Tour



- 1 - Show Annotation Tools
- 2 - Participant Panel
- 3 - Chat Panel
- 4 - Reactions/Raise Hand
- 5 - Mute/Unmute

This slide can be used to introduce students to features used in this course that they might have not used before, such as annotations, hand raising, or chat.

Introductions

- Name
- Company or Industry
- Job Role
- Atlassian Products?
- Fun Fact



Please share in chat and select Send to Everyone

This slide can be used to introduce your students to each other to make them more comfortable with interacting.

Where in the World?

Use your annotations to highlight your location!



This slide further familiarizes students with each other and starts a dialog.

What is your experience with Jira Software?

Limited
Experience

Some Experience

Extensive
Experience



Experience indicators help the instructor gauge the student's knowledge of the material in the course. For example, if everyone has extensive experience, the instructor may gloss over the more straightforward subjects. Instructors can also identify students with limited experience who may need extra help.

Lab 1 – Course Overview

- Log in to Jira



2

Agile and Jira Overview



In this video we will discuss both agile and Jira overview.

What will you learn?



- Describe agile
- Describe Jira
- Identify how Jira relates to an agile mindset
- Create a Jira issue
- Use a Jira project
- Use a project board
- Identify Jira user types



We will describe agile and Jira, identify how Jira relates to an agile mindset, create a Jira project, create a Jira project or an issue, use a project board, and identify all Jira user types.

Topics

Agile overview

Jira overview

Projects, issues, and boards



Let's start with an agile overview.

What is agile?

A way of getting things done.

An empirical approach to project management

Continuously develop the plan, process and product

A mindset



What is agile? If you asked ten people, you would get 10 different answers.

Some examples - it's a way of getting things done.

It's an empirical approach to project management - you use a scientific or data-driven approach.

With agile, you continuously develop the plan, the process, and the product. This is in contrast with developing the processes and plan up front and then executing the creation of the product in stages.

It is also a drill - a mindset or a way of thinking. That's a big focus in this course to understand that agile mindset.

Agile is ..

Project management approach

Agile is a project management approach.

Breaking down work

It involves breaking the project into phases.

Continuous improvement

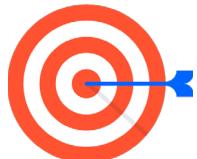
Teams follow a cycle of planning, executing, and evaluating.



The Agile methodology is a project management approach that involves breaking the project into phases and emphasizes continuous collaboration and improvement.

Agile teams follow a cycle of planning, executing, and evaluating work.

Why agile?



Effectiveness
Perform better than traditional projects



Empower the Team
Leverage team knowledge and increase job satisfaction



Manage Complexity
Simple project management approach to increasing complexity



Why are teams using Agile approaches?

First, because of its effectiveness. In many contexts, teams that work in an agile way are more effective than teams that work in traditional methods such as waterfall methodology.

. An agile team can often build a product faster, more reliably, and at a lower cost than conventional teams.

Agile companies empower their agile team members. Team members are often in the best position to make certain decisions, such as how to get the work done—an empowered person is generally happier than an un-empowered person, which leads to higher overall job satisfaction.

As project complexity grows, using a complex project management approach adds to the complexity and decreases the chance of project success. Agile approaches to project management are relatively simple, making the overall project complexity much more manageable.

What is an agile mindset?

- A growth/continuous improvement way of working
- Allowing the data to change your approach
- Uses agile techniques to accomplish work



What is an agile mindset?

It's a growth or continuous Improvement way of working instead of having fixed skills and processes.

You are continuously improving an agile mindset means that you are open to empirical approaches, and this means that you're allowing the data to change your approach.

And finally, an agile mindset uses agile values, principles and techniques to accomplish work. We will see many of these techniques throughout this course.

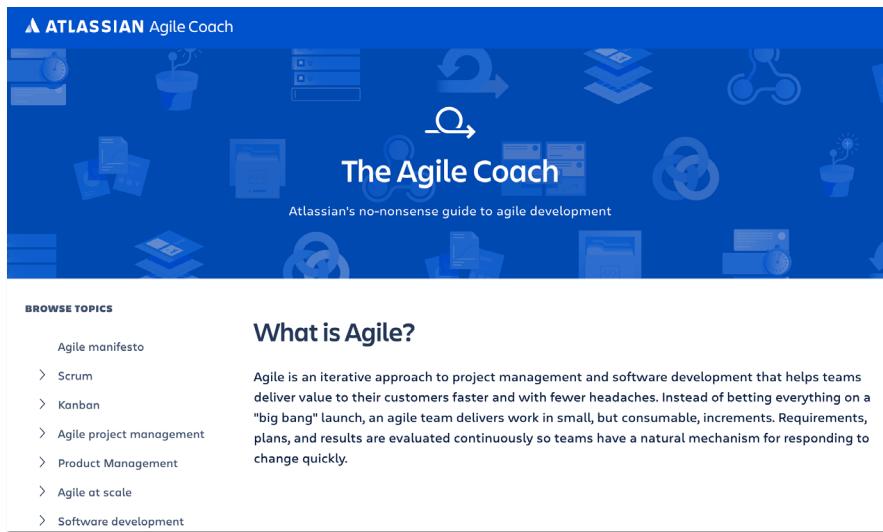
Why an agile mindset?

For an agile team to perform its best, all team members must have an agile mindset



So why is it important to have an agile mindset for an agile team to perform at the best. A team cannot just be told to “be agile”. For a team to achieve optimal effectiveness, every member of the organization, including leaders and team members, must have an agile mindset.

The agile coach (atlassian.com/agile)



The screenshot shows the homepage of the Atlassian Agile Coach site. The header features the Atlassian logo and the title "The Agile Coach". Below the header is a banner with the text "Atlassian's no-nonsense guide to agile development". The main content area has a section titled "BROWSE TOPICS" with links to various agile topics like Agile manifesto, Scrum, Kanban, etc. To the right, there is a large section titled "What is Agile?" with a detailed description of the Agile methodology.

ATLASSIAN Agile Coach

The Agile Coach

Atlassian's no-nonsense guide to agile development

BROWSE TOPICS

- Agile manifesto
- > Scrum
- > Kanban
- > Agile project management
- > Product Management
- > Agile at scale
- > Software development

What is Agile?

Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches. Instead of betting everything on a "big bang" launch, an agile team delivers work in small, but consumable, increments. Requirements, plans, and results are evaluated continuously so teams have a natural mechanism for responding to change quickly.



The agile coach site (atlassian.com/agile) contains great resources related to many aspects of agile. You are encouraged to take a look!

Topics

Agile overview

Jira overview

Projects, issues and boards



Next, we will discuss Jira Overview.

What is Jira?



Plan and Track

Plan, track and work on work items.



Visualize and report

See the progress of work and view reports.



Processes

Implement processes that match your team.



Jira is a software that teams can use to plan, track and report on work while following agile principles. Jira provides tools that can help teams with every stage of the project lifecycle. For work visualization, teams can use boards, dashboards and built in reports. This can also help make better decisions for the project and team when needed.

Additionally, Jira is a highly configurable and customizable tool. It allows teams to design their processes within the tool and adapt it to their work rather than having to adapt themselves to Jira.

Why use Jira?



Save time and energy

Jira provides project management capabilities to help teams save time and focus on actual work.



Work better

Jira helps plan, prioritize, visualize, and complete work



Team communication

Jira allows teams to communicate more efficiently.



So why use Jira?

Jira helps teams save time and energy to focus on actual work, by providing project management tools such as backlogs, time tracking and roadmaps.

Also, using Jira, teams can plan, prioritize, visualize, and complete work. They can use boards and dashboards, plan sprints and view charts.

Finally, Jira boosts team communication via comments and notifications.

How does Jira relate to an agile mindset?

Jira features are designed for agile teams, allowing them to model and execute agile processes.



How does Jira relate to an agile mindset? Jira is a tool teams can use to model and execute their agile processes as well as help implement the practices related to an agile mindset.

Topics

Agile overview

Jira overview

Projects, issues, boards, and user types



Next, we will discuss projects, issues, boards, and user types.

What is a Jira issue?

An item of work

An issue is an item of work defined by the team.

Issue type

An issue has an associated type defining what it represents (e.g. Story, Bug).

Fields

Issues have fields that hold information about the work item (e.g. assignee).



In Jira, we refer to teams' work items as issues. The term “issue” comes from Jira’s historic roots as bug, or issue tracking software.

Every issue has an associated type. Issue types include stories, tasks, bugs, epics and custom issue types that Jira administrators can create. Each issue type can have a workflow associated to it defining its lifecycle.

Also, every issue holds that is broken down into fields. For example, the assignee field holds the user working on the issue, and the summary field holds a brief text summarizing what the work item is about.

Custom fields can also be created to match your team's processes.

What is a Jira project?

Collection of issues

A project is a collection of related issues.

“To do” list

It can be perceived a team’s to do list.

Project type

Can be Team-managed or Company-managed.

Project template

Pre-configured basic projects (e.g. Scrum or Kanban project)



A Jira project doesn’t necessarily have a start and end date.



A Jira project is a collection of related issues. Projects help organize work.

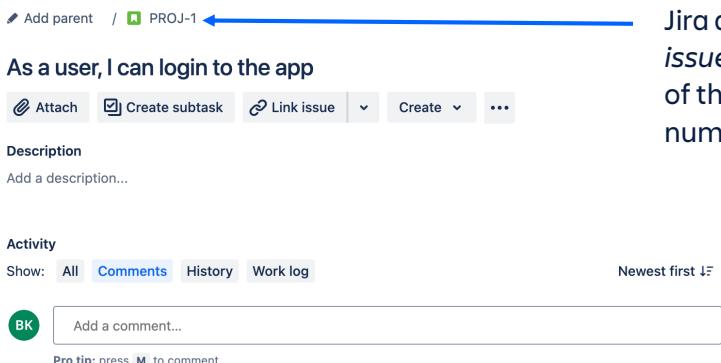
You can think of the issues in a project as the team’s “to do” list. The issues can be in different statuses, such as In Progress, Under review or ready for development.

A Jira project is not necessarily a project in the traditional sense, which usually has a start and end date. The term “project” is used loosely to include work that has no planned end date. For example, a product that constantly needs to be improved and released.

Projects can be of two types: Team-managed or Company-managed. The main difference is in how the projects are configured and who can do it. Team-managed projects are easily configurable projects that can be created by regular users. Whereas company managed projects can only be created by Jira admins and their configuration is very dependent on the Jira admin as well.

However, our focus in this course is software projects. When you are creating a software project, you have to select a project template that defined the minimum configuration of the project. For example, when selecting the scrum template, you will get an associated backlog and scrum board.

Issue key



The screenshot shows the Jira interface for creating a new issue. At the top, there's a breadcrumb navigation: 'Add parent / PROJ-1'. Below it, a heading says 'As a user, I can login to the app'. A toolbar contains buttons for 'Attach', 'Create subtask', 'Link issue', 'Create', and more. The main area has a 'Description' section with a placeholder 'Add a description...'. Below that is an 'Activity' section with tabs for 'All', 'Comments' (which is selected), 'History', and 'Work log'. It shows 'Newest first' activity. A comment input field says 'Add a comment...' with a note 'Pro tip: press M to comment'. A small circular icon with 'BK' is visible.

Jira automatically assigns a unique **issue key** to created issues, composed of the project key (PROJ) and the issue number (1).



Jira automatically assigns a unique issue key to each issue. The letters before the dash represent a unique identifier for the project. This is called the *project key*. The issue number in the project follows the dash. Issue key values are unique to the Jira account. To ensure this, you can not have two projects with the same project key.

Each issue belongs to one project



Every issue in Jira is unique and belongs to just one project. In this example, we have two projects: iOS App and HR projects.

The issues are the project's work items. For instance, the HR Onboarding project may have issues such as create email account, supply laptop and assign desk. The iOS project may have issues such as code login button, create help screen and configure iCloud.

An issue can only belong to one project.

What is a project board?



To do list

A two-dimensional “to do” list.



Visualize

Boards help visualize work and its progress throughout the process.



Cards

Displays issues as cards

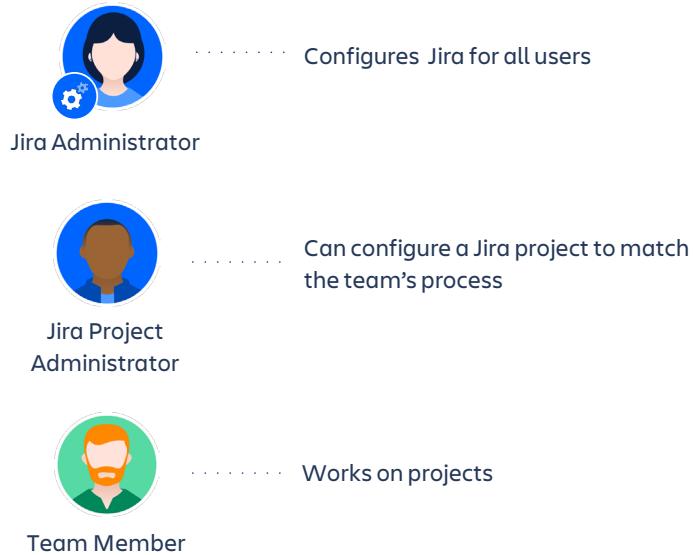


A project board is a two-dimensional view of the work to be done by the team. Work items can be in multiple states and be worked on by multiple team members.

Project boards also help visualize the work as it progresses throughout the defined process.

Boards present issues as cards, and each card displays issue fields for easy visualization. The fields displayed on the issue cards can be configured by the team, depending on what information they consider as key.

Types of users in Jira



There are three main types of Jira users:

A *Jira administrator* configures the Jira instance for all users. They in general know the most about the technical capabilities of Jira and can set policies for the entire company to use with their Jira projects. The changes that they can make can affect multiple projects, so they must be very knowledgeable of Jira.

A *Jira project administrator* can configure a specific project to match the team's desired process. Jira project administrators work closely with the agile team to understand the desired process, and works with the Jira administrator when they do not have permissions to perform some of their desired Jira configuration.

A team member uses Jira to work on projects.

In general, a company has a few Jira administrators, more Jira project administrators, and even more team members.

Are you getting it?



1. Which one of the following statements about a Jira project is true?
 - a. A Jira project is used to track bugs only.
 - b. A Jira project contains related issues. Issues can be of type story, task, or bugs.
 - c. A Jira project contains issues of type stories only.



The answer is on the next slide.

Did you get it?



1. Which one of the following statements about Jira project is true?
 - a. A Jira project is used to track bugs only.
 - ✓ b. A Jira project contains related issues. Issues can be of type story, task, or bugs.
 - c. A Jira project contains issues of type stories only.



Answer: b

Takeaways



- Agile is a way of working
- Jira is a tool teams use to manage and visualize work
- Jira can be configured to match a team's continuously improving processes
- A Jira issue is an item of work identified by the team
- Project boards visualize a team's work
- The main types of Jira users are Jira administrators, Jira project administrators and team members



Here's a review of what we discussed.

Lab 2 – Agile and Jira Overview

- Explore a kanban project
- Create issues
- Star items



3

Visualize Work Using Project Boards



What will you learn?



- Describe the importance of visualizing work
- Describe common workflows
- Differentiate Jira boards and workflows
- Describe the purpose of an issue's status field



Topics

Visualizing work

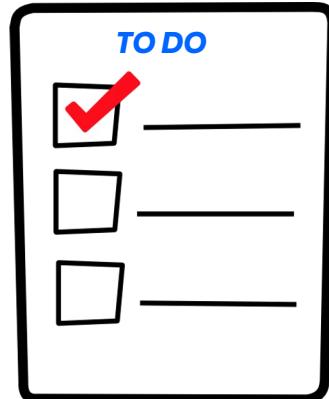
Workflows

Jira boards and workflows



Visualizing work in a “to do” list

- Reminds you
- Focuses you
- Sets priorities
- Tracks progress



A to do list is a classic example of a tool used to visualize your work. Even though a to do list is very simple to use, it has some valuable characteristics.

It visually reminds you of the work that you need to do. We all have a tendency to forget things if we don't write them down, especially if there are a lot of items. A to do list helps prevent you from forgetting things.

A to do list focuses you, because you can concentrate on doing only the things on the list.

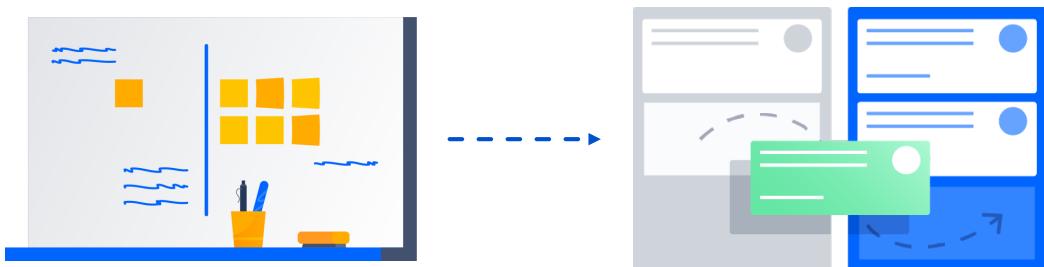
If you want, you can prioritize the work items on your to do list, simply by changing their order.

As you check off your work items, you are tracking your progress. This is usually quite rewarding.

We will see that this simple example of a to do list contains a lot of the building blocks related to visualizing the work of agile teams.

Visualizing work in a board

- A principle of agile projects is to "visualize work"
- A board is an agile tool used to visualize and manage work



A principle of agile projects is to visualize work. A board is an agile tool used to help visualize and manage the work of the team. Depending on the circumstances, a board may also be referred to as a task board, a project board, a kanban board or a scrum board.

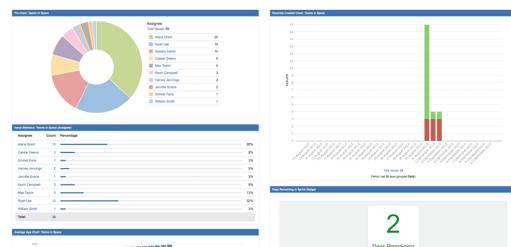
A board can be a physical board like a whiteboard with sticky notes, or can be based in software. Depending on which template that you choose when creating the project, a board can be automatically created for you.

A board contains columns. Each column can contain issues, which are the work items for the project. A board is like a two-dimensional to do list. The extra dimension is used to allow a work item to go through multiple steps by multiple team members before it is finished. Agile teams commonly break up the project into manageable issues and work on them in steps like this.

Visualizing work: reports and dashboards



Reports



Dashboards



We saw that boards are used to visualize the work of the team. In Jira, reports and dashboards are automatically created and updated for you. This is another great way to visualize the team's work, especially for identifying problems.

Why visualize work?

- To easily **see** the work of the project
 - Allows anyone to see the true current state of the project
 - Organizes and focuses the team
- To **manage** things
 - Easy to add and prioritize the work of the project
 - Easy to update work items
- To **improve** the team's way of working
 - Can visually identify problems



Why is visualizing work an important principle of agile? Like a to do list, visualizing work with tools such as a board makes things easier, better and more rewarding.

Visualizing work allows everyone to see the current state of the work of the project. With a board, you can easily see which items are done, which items are in progress, and which items have not been started. Boards are very transparent in that they show the true state of the project, not only to the project team, but to any stakeholders who have access to the board.

Visualizing work also organizes and focuses the team. The team should only be working on the issues on the board, and the next issue to be worked on is obvious.

Visualizing work allows you to manage things. For example, the team can easily add work items to the board, modify existing work items or change their priority. As the team works on a work item, the item can be updated to show the progress.

Visualizing work can also improve the team's way of working. A key principle of agile projects is to continuously improve, not only the product, but the way that the team works on the product. A board can help the team visually identify problems or bottlenecks with the process. This highlights areas for the team to focus on improving.

Topics

Visualizing work

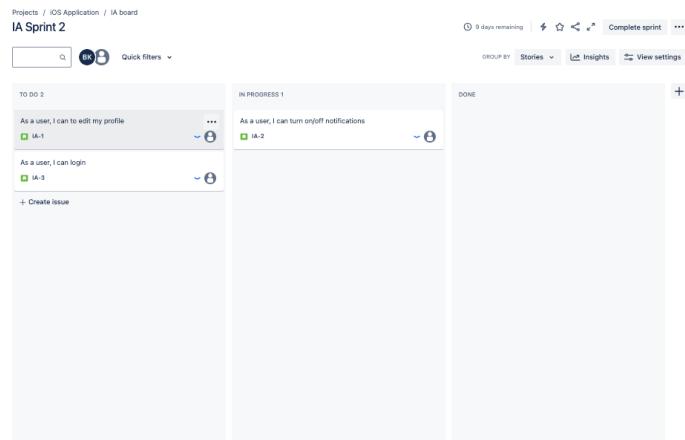
Workflows

Jira boards and workflows



Workflows

Columns in a board represent a *workflow, which is broken down into statuses (e.g. To Do).*



The set of columns of a board represent a workflow for completing an issue. Workflows are used to model the processes involved in the project. Even though a workflow technically can be considered a model of a process, you often will find the terms workflow, process and business process used interchangeably. They all represent breaking down the work into a series of steps.

Each column in a board usually represents a single step in the workflow. These steps may also be called statuses.

You can see that workflows and boards are closely related. The board visualizes the workflow.

The board shown here has four statuses in the workflow. These statuses happen to have the same name as the columns in the board. The statuses are backlog, selected for development, in progress and done.

Use case : order and delivery process



Let's take a simple process and model it as a workflow. We will model the process of ordering and delivering a customer's food at a restaurant.

The first step of the workflow is for the wait staff to take the order from the customer.

In the second step, the wait staff adds the order to the cook's queue. It's placed in a queue because the cook only can start the order when they have bandwidth.

In the third step, the cook takes the order from the incoming queue and begins to prepare the order.

In the fourth step, the cook has finished the preparation and adds the order to the delivery queue. It's placed in a queue because the cook usually doesn't directly deliver the order to the customer and the wait staff might not be immediately available.

In the final step of the workflow, the wait staff delivers the order to the customer.

Even though this is a simplified example of a workflow, the same basic idea of breaking down the work of the project into steps applies. One of the benefits of breaking down the workflow into steps is that the work of the steps can be done by different performers, allowing the process to scale to large teams. You can model any process that you want using this simple block diagram approach. All it takes is a pencil and paper.

Boards vs. workflows

The team board structure is defined by the underlying workflow.

The screenshot shows a Jira board titled "Orders board". The board has five columns: "ORDER CONFIRMED 2", "COOKING QUEUE", "PREPARE ORDER 1", "DELIVERY QUEUE", and "DONE". The "ORDER CONFIRMED 2" column contains two items: "Order 3" and "Order 2", both with status "IA-1" and "IA-3" respectively. The "COOKING QUEUE" column is empty. The "PREPARE ORDER 1" column contains one item: "Order 1" with status "IA-2". The "DELIVERY QUEUE" and "DONE" columns are also empty. A red dashed box highlights the first column, "ORDER CONFIRMED 2".



A board's structure is defined by an underlying workflow. In this example, we can see that each column in the board represents a status of the orders workflow we defined in the previous slide.

The board columns are configured by the board administrator to match the workflow, it is not automatically configured.

Topics

Visualizing work

Workflows

Jira boards and workflows



How are boards created?

- A board is automatically created with software projects
- Create additional boards at any time

The screenshot shows a Jira Kanban board titled "Orders board". The board has four columns: "ORDER CONFIRMED 2", "COOKING QUEUE", "PREPARE ORDER 1", and "DELIVERY QUEUE". The "ORDER CONFIRMED 2" column contains two items: "Order 3" and "Order 2", each with sub-tasks "IA-1" and "IA-2". The "PREPARE ORDER 1" column contains one item: "Order 1". The "DELIVERY QUEUE" column is empty. In the top right corner, there is a context menu with the following options: "Configure board", "Create board" (which is highlighted with a red arrow), "Export (PDF HTML Word)", and "Export PDF".

Balls are automatically created when you create a project using kanban or scrum templates. In this example, we have created a kanban project.

You can create additional boards at any time. A single project can have multiple boards, and a single board can contain the issues of multiple projects. (Currently, you can not create extra boards with cloud team-managed projects.)

An issue's status field

- Every project automatically has one or more associated workflows
- The status field for each issue must be set to a workflow's status

The screenshot shows a Jira issue page for an issue identified by the key SKP-5. The title of the issue is "Login page is blank". Below the title, there are buttons for "Add parent", "Create subtask", "Link issue", "Create", and a three-dot menu. The "Status" field is currently set to "Backlog". A red arrow points to the "Actions" dropdown menu, which is open and displays a workflow status selection interface. The interface includes a table with columns for "Status" and "Assignee". The rows show "SELECTED FOR DEVELOPMENT" (status: IN PROGRESS, assignee: alikis Khouni), "IN PROGRESS" (status: IN PROGRESS, assignee: alikis Khouni), and "DONE" (status: DONE, assignee: alikis Khouni). There are also buttons for "View workflow", "Development", and "Create branch".

Every issue type in a project is associated to a workflow. Issues with that issue type will follow the associated workflow.

The status field defines the workflow status for an issue.

For instance, let's consider this issue identified by the key SKP-5. The status of the issue is set to "Backlog," and the dropdown menu provides three alternative statuses that we can set for this issue.

Boards and status

The screenshot shows a Jira board titled "Orders board". The board has five columns: "ORDER CONFIRMED 2", "COOKING QUEUE", "PREPARE ORDER 1", "DELIVERY QUEUE", and "DONE". The "ORDER CONFIRMED 2" column contains two issues: "Order 3" (IA-1) and "Order 2" (IA-3). The "PREPARE ORDER 1" column contains one issue: "Order 1" (IA-2). The "DONE" column has a link to "See older issues". The top right of the board includes buttons for "Release", "GROUP BY", "Queries", "Insights", and "View settings".

VIEW ISSUES

Boards are a view of issues grouped by status.

MOVING ISSUES

Moving an issue within the board changes the value of its status.



We are now in a better position to understand boards. Boards are a view of issues grouped by status. When you move an issue to a different column in the board, you are changing the value of its status field. Similarly, when you view an issue's details and change its status, you will usually see the issue change columns on the board.

Are you getting it?



1. Why are there multiple columns in an agile board?
 - a. One column for each team member.
 - b. One column for each work item.
 - c. One column for each status that a work item goes through.



The answer is on the next slide.

Did you get it?



1. Why are there multiple columns in an agile board?
 - a. One column for each team member.
 - b. One column for each work item.
 - ✓ c. One column for each status that a work item goes through.



Answer: c

Are you getting it?



2. On a board, what happens when you move an issue from one column to another?
 - a. The issue's workflow changes.
 - b. The value of the issue's "status" field changes.
 - c. A new issue is created.



The answer is on the next slide.

Did you get it?



2. On a board, what happens when you move an issue from one column to another?
 - a. The issue's workflow changes.
 - ✓ b. The value of the issue's "status" field changes.
 - c. A new issue is created.



Answer: b

Takeaways



- A board is a two-dimensional way to visualize the work of a team
- In Jira, a workflow is often represented using a board
- Board columns usually map to the *status* field of issues
- Board columns can be added or removed to match the team's desired process



Lab 3 – Visualize Work Using Boards

- Move issues through a workflow
- Assign an issue



4

Enrich issues



What will you learn?



- Identify ways that issues can be enriched with information
- Describe the benefits of using issue types
- Describe subtasks
- Use labels to organize issues
- Introduce integration with version control and build systems



Topics

[Enriching issues](#)

[Issue types](#)

[Labels](#)

[Developer integration overview](#)



Issues contain work-related information

Issue

| | | | |
|--------------|---|---|--|
| Summary: | Check network jacks | | |
| Description: | Each network jack in the new building needs to be checked for signal strength. | | |
| Type: |  Task | Assignee: |  Helena |
| Priority: |  Critical | Reporter: |  Oliver |
| Status: |  IN PROGRESS | Comments: Helena needs the network diagram from IT. | |



An issue in Jira can contain all the information about that particular work item in one place. This information is stored in fields.

The assignee is the person currently assigned to work on the issue. This often changes throughout the life of the issue.

The reporter is usually the person who created the issue. An issue will only have one reporter and this rarely changes.

The fields you see in an issue depend on the type of issue and how your project is configured. You can also create custom fields to match your team's process.

Enriching issues

ATTACH SCREENSHOTS

Users can attach images to issues.

LOG WORK

Users can log time against issues they worked on.

LINKING

Users can link related issues.

ESTIMATES

Users can put estimates on issues to predict completion time.



Here are some other common actions you can perform to enrich issues:

- You can add **attachments** to issues, such as a **screenshot** showing an error or an **invoice** file. This keeps all the important information easily accessible for the issue.
- You can **link** an issue to another one. For example, a problem request in an IT Service Desk can be linked to the bug that caused it in Jira Software. By linking these two issues, the support engineer can easily navigate to the bug issue to see what's being done. Also, the problem request will automatically be updated when the bug is fixed.
- When you create an issue, you can estimate the **time** needed to complete it.
- As you work on an issue, you can **log the time** and the actual **work** performed on the issue. This is important for project planning, tracking and reporting.

Mention team members

Issue

Summary: Add artwork to company website

Type: Task

Assignee:



Helena

Priority: Critical

Status: IN PROGRESS

Comments: Helena needs to get the artwork files from

@alex.

@Mention
Alex Dupree



Alex Dupree



You can “at mention” (@mention) team members in an issue, and they will be notified that they have been mentioned.

By simply entering the @ character, a list of available team members will appear for you to select. Jira will then notify that team member that they have been mentioned in a particular issue.

Helena is assigned to add some new artwork to the company website. She needs to get the files from Alex. Helena enters @Al and Alex’s name appears in the list for Helena to select. Helena has now tagged Alex to the issue by using an at mention.

Topics

Enriching issues

Issue types

Labels

Developer integration overview



The issue type field

- **Epic** - a big issue that can contain child issues
- **Story** - requirement from the user's perspective
- **Task** - team work item
- **Bug** - a flaw that needs to be fixed
- **Subtask** - a child issue

Create issue

Project*
projectA (PROA)

Issue Type*
 Story

Task
 Bug
 Epic



A project can also use custom issue types created by admins



In Jira, an issue is a generic name for a unit of work. On your projects, there usually are different types of units of work. The issue type field is used to differentiate these different types of units of work. When you create an issue, you must choose the issue type. You can also change the issue type after creating the issue. Notice that each type has an associated icon to help easily identify the issue type.

A story is a requirement from the user's perspective.

A task is usually a work item that needs to be done by the team but is not directly tied to a user requirement. An example might be upgrading the version of a product used by the team.

A bug is a flaw that needs to be fixed in the product. It can be tracked with its own issue type to differentiate it from other types of work as it can affect the delivery date.

An epic is a big issue that can contain other issues.

A subtask is a child of another issue. It is used to break an issue down into specific pieces of work. When creating an issue, subtask is not shown in the issue type dropdown because subtasks must have a parent issue- they can not be created independently.

In addition to these out of the box issue types, an admin can create custom issue types. This provides your team the flexibility to work the way that they want to work. How the team actually defines what each issue type means is entirely up to them. Also, different projects can use different issue types.

Why issue types?

DIFFERENTIATE WORK ITEMS

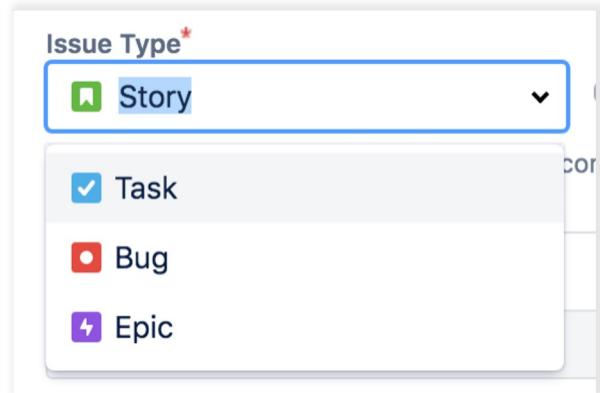
Issue types help differentiate different types of work items.

FIELDS

Each type can have different fields, screens and workflows.

REPORTING

Can report on types separately



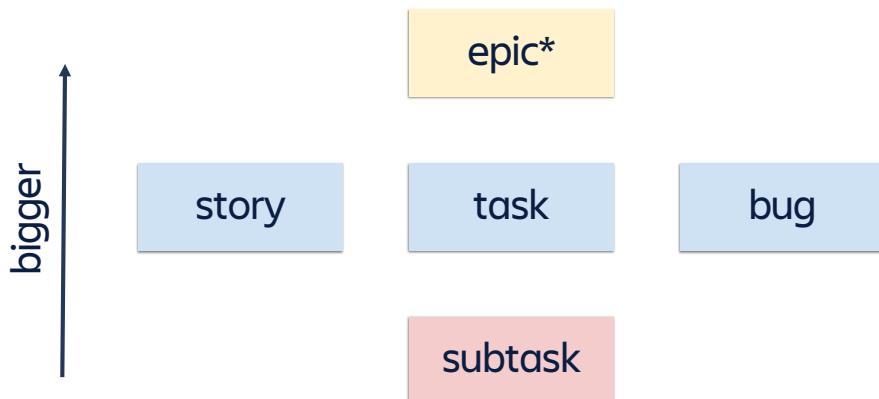
There are several reasons to use issue types with your projects.

Issue types support different types of work items. A team usually doesn't have only one type of work, and issue types allow the team to differentiate those types of work.

Each type can have different fields, screens, and workflows. For example, you might want bugs to appear at the top of the project's board.

You can also report on types separately. For example, because the issues of the project have been categorized by issue type, you can easily create a report with the number of bugs fixed in the previous week.

Jira's issue type hierarchy



* Epics are discussed later



Issue types in Jira follow a hierarchy, and depending on which level of the hierarchy they are in, they might have specific features.

In the default configuration of Jira, you have three levels of hierarchy:

Epics represent a large chunk of work that can be broken down in multiple work items called child issues. With epics, you have a dedicated button for creating child issues in the issue detail view.

The second level, under an epic, can be a story, a task or a bug. The issue types in this level represent standard work items that can be worked on by one individual.

Subtask is the last issue type level in the hierarchy. A subtask has to have a parent, it cannot be created by itself. Additionally, it is not possible to create a child issue under a subtask.

Jira administrators can create custom issue types that fit in the hierarchy. They can create standard issue types or subtask issue types. They can grant them any name they like (example: Incident).

EPIC

Big work item

Epics represent a large piece of work that can be broken down into multiple issues.

Create child issues

You can create child issues from the epic detail view.

List of child issues

Child issues are listed in the epic, making it easy to have visibility over the progress of the epic.

| Issue | Description | Order by |
|--------|---------------------------------|----------|
| TRIP-5 | As a user I can login | 1 |
| TRIP-6 | As a user I can edit my profile | 2 |
| TRIP-7 | As a user I can book a trip | 3 |

Epics usually represent a large piece of work that can be broken down into child issues, typically stories. You can use the Add a child issue button to create a new child issue under an epic. The child issues will be listed in the epic detail view which provides visibility over the progress of the epic.

SUBTASK

Has a parent

An issue type that must have a parent issue

Break down work

Allow an issue to be broken down into individually manageable tasks

Specific task

Can be more technical than the parent issue

The screenshot shows a Jira issue page for an issue titled "Check network jacks". The status is "In Progress". On the left, there's a sidebar with sections for "Details", "Assignee" (Adam), "Reporter" (Adam), and "Development" (Create branch, Create commit). The main area has tabs for "All", "Comments" (which is selected), "History", "Work log", and "Newest first". A "Create a subtask" button is highlighted with a blue arrow pointing from the "Has a parent" section above. Below it is a "Description" field with placeholder text "Add a description...". At the bottom, there's a comment input field with the placeholder "Add a comment..." and a tip: "Pro tip: press M to comment".

Subtask is an issue type that must have a parent type.

Subtasks allow an issue to be broken down into individually manageable tasks. They can be assigned to different team members.

Subtasks can be more technical than the parent issue. For example, if the parent issue is a story, the story will be written in non-technical language that all team members and stakeholders understand, but the subtasks can be written for the technical person implementing the subtask.

The create subtask button is available on every issue with a standard issue type, unless the Jira admin has restricted the creation of subtasks through project permissions.

SUBTASK

Issue key

Have their own issue key and field values.

Workflow

Subtasks can have dedicated workflows.

The screenshot shows a subtask details page. At the top, there's a breadcrumb navigation showing 'Add parent / IT-1'. Below it is the subtask title 'Check network jacks'. To the right, there's a status indicator 'In Progress' with a dropdown arrow, also highlighted with a red box. On the far right, there's a 'Actions' button. The main content area includes a 'Description' section with a placeholder 'Add a description...', an 'Activity' section with tabs for 'All', 'Comments' (which is selected and highlighted with a blue box), 'History', and 'Work log', and a sorting option 'Newest first ↓'. Below the activity section is a comment input field with a placeholder 'Add a comment...' and a pro tip: 'Pro tip: press M to comment'. To the right of the main content, there's a sidebar with user details: 'Assignee' (Adam), 'Reporter' (Adam), and development-related links: 'Create branch' and 'Create commit'.

Subtasks have their own issue keys and fields. Here is a story with two subtasks. Each subtask has its own issue key and summary field value.

The subtasks have independent workflow statuses and move through boards independently.

Issue types affect where issues are listed



Epics

Can be viewed in the epic panel of the backlog or in the Epic link field to indicate the relationship of a child issue to an epic.



Tasks, bugs and stories

Issues with standard issue types are typically viewed in the backlog. They are also listed within the parent epic.



Subtasks

Subtasks are listed within the parent issue detail view.



Depending on their issue types, issues can appear in different ways and locations.

Epics for example can be viewed in the epic panel in the backlog of the software project.

Issues with standard issue types such as Tasks or stories or bugs can be viewed in the backlog or listed in their parent epics.

Subtasks however, are listed within their parent issue for better visibility.

Topics

Enriching issues

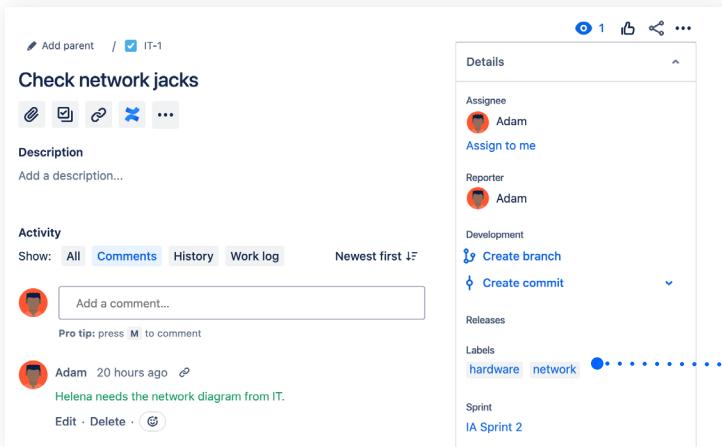
Issue types

Labels

Developer integration overview



Labels

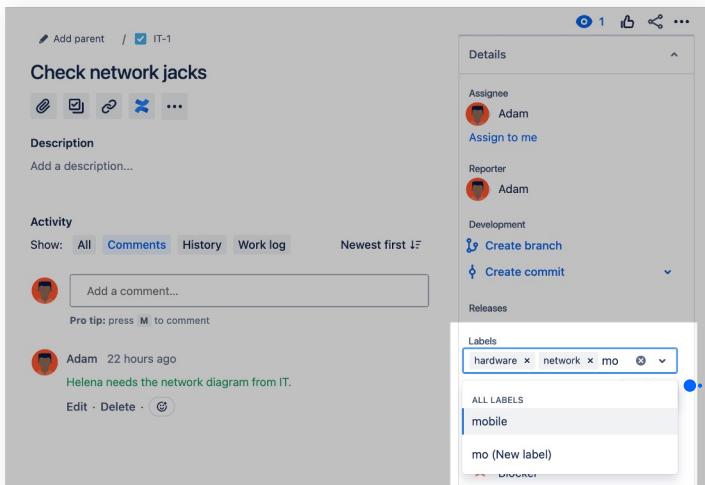


The screenshot shows a Jira issue page for a ticket titled "Check network jacks". The ticket has two labels: "hardware" and "network". The right side of the screen displays the "Details" panel, which includes fields for Assignee (Adam), Reporter (Adam), Development (Create branch, Create commit), Releases, Labels (hardware, network), and Sprint (IA Sprint 2). A blue callout box points to the "Labels" section of the Details panel with the text: "The Labels field is used to categorize and search for issues".



What if you have issues in a project that relate to the database, or apply to different types of work within the project? What if you have information about an issue that doesn't really fit in a field, but you want to track it so you can search and report on it? Labels are a field used to help categorize and search for issues in any way that makes sense to the team. You can have multiple labels per issue. This issue has two labels- refactor and database.

Adding or creating a label



You can reuse labels or create new ones.



Labels can be added as you create an issue or can be added to existing issues. Jira will suggest existing labels as you type. In this example, in the create issue screen, we typed "mo", and we see that we can select the mobile label. We could also type a new name and the label will be created.

Searching for labeled issues

Click on a label to search for all issues with this label

The screenshot shows a Jira interface with the following details:

- Header:** Your work, Projects, Filters (selected), Dashboards, Teams, Plans, More, Create, Search bar.
- Toolbar:** Apps, Share, Export issues, List View, Detail View (selected), ID, BASIC, JQL.
- Search Bar:** labels=network
- Issue List:** One issue titled "Check network jacks" is listed. It has a status of "In Progress".
- Issue Details:** Description: "Add a description...", Activity: Show: All, Comments, History, Work log, Newest first. Comment: "Adam yesterday Helena needs the network diagram from IT.".
- Labels:** hardware, network (highlighted with a blue arrow).



You can click on a label in an issue, and Jira will take you to a search page containing all issues with this label. In this case, two issues use the “hardware” label. You can also search for issues with certain labels using basic search or JQL, which we will discuss later.

Topics

Enriching issues

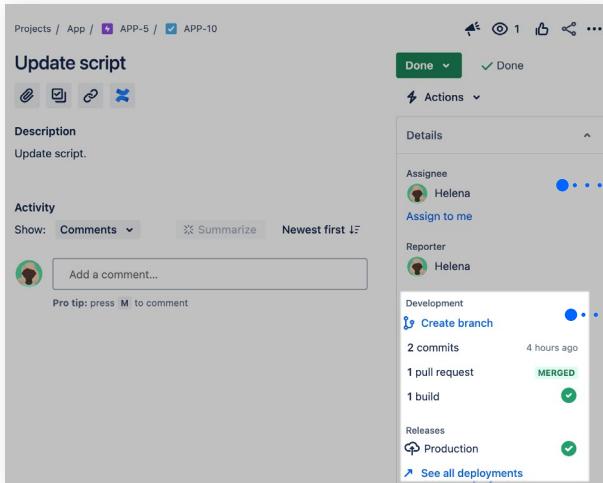
Enriching issues

Labels

Developer integration overview



The issue detail development panel



The screenshot shows the Jira Software interface for an issue titled "Update script". The main area displays the issue's description: "Update script." Below it, the "Activity" section shows "Comments" (with a dropdown menu), "Summarize", and "Newest first". A comment from user "Helena" is visible. To the right, a "Details" sidebar is open, showing "Issue context fields" (Assignee: Helena, Reporter: Helena) and a "Development" section. The "Development" section lists: "Create branch" (4 hours ago), "2 commits" (4 hours ago), "1 pull request" (MERGED), "1 build" (green checkmark), and "Releases" (Production). A blue callout points to the "Development" section with the text: "Commits, pull requests and builds related to the issue. This information is pulled from connected repositories."

Issue context fields

Commits, pull requests and builds related to the issue. This information is pulled from connected repositories.



When Jira Software is integrated with development tools, the development panel is displayed with the issue details.

In the development panel, there are two commits related to this issue, one pull request that's been successfully merged, and one successful build. This issue was included in a deployment to Production.

Integration works through the issue key

USING A COMMIT MESSAGE

Include an issue key in a commit message
“Initial commit – TIS-498”.

USING A BRANCH NAME

Include an issue key in a branch name
“Feature branch TIS-498”.

FOR PULL REQUESTS

Include an issue key in a pull request title,
or Jira can use the issue key from an
associated commit or branch.

FOR BUILDS AND DEPLOYMENTS

Jira uses the issue key associated with a
commit in the build.



The integration is enabled through a reference to the issue key. This reference can be in a commit message or a branch name.

For pull requests, the issue key can be used in the pull request title, or the pull request can just get the reference from a commit or branch associated with the request.

Builds and deployments use the issue key reference from an associated commit.

Are you getting it?



What are some of the ways you can enrich an issue?

- a. Attach screenshot
- b. Link issues
- c. Log work
- d. Estimate completion time
- e. All of the above



The answer is on the next slide.

Did you get it?



What are some of the ways you can enrich an issue?

- a. Attach screenshot
- b. Link issues
- c. Log work
- d. Estimate completion time
- ✓ e. All of the above



Answer: e

Are you getting it?



Which of the below is true for Labels? Choose all that apply.

- a. Labels are a field used to help categorize and search for issues in any way that makes sense to the team.
- b. You can have only one label per issue.
- c. You can have multiple labels per issue.



The answer is on the next slide.

Did you get it?



Which of the below is true for Labels? Choose all that apply.

- ✓ a. Labels are a field used to help categorize and search for issues in any way that makes sense to the team.
- ✓ b. You can have only one label per issue.
- ✓ c. You can have multiple labels per issue.



Answer: a and c

Takeaways



- An issue contains a diverse set of fields that are used to add information to the issue
- Issues can facilitate team communication with comments and @mentions
- Issue types can have unique fields, screens and workflows
- Subtasks are children of another issue type
- Subtasks have their own issue key and field values
- Labels can be used to categorize and search for issues
- Jira can be integrated with version control and/or build systems to improve developer-related communication



Lab 4 – Enrich Issues

- Add information to an issue
- Use team-related issue features
- Create issue of different types
- Create subtasks
- Add labels to issues



5

Kanban Method



What will you learn?



- Describe the kanban method
- Describe the importance of flow
- Identify the purpose of work-in-progress limits
- Differentiate pull vs push processes



Topics

Kanban method overview

Improving flow

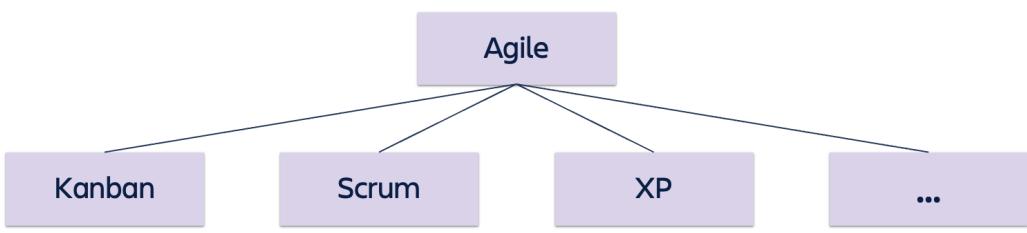
Pull vs push



We'll start with the Kanban method overview.

Agile methods

- Agile is a way of thinking (mindset) and working
- An agile method (or framework) is an approach to implementing agile
- Common agile methods include kanban and scrum
 - Each embodies core principles of agile
 - Often combined



Agile methods are approaches to achieving agility. The methods are also sometimes called frameworks or methodologies. Agile on its own is more of a mindset than an actual method for managing projects. Agile methods add structure to agile ideas.

Common agile methods are kanban and scrum. There are others as well, such as XP, or extreme programming.

Even though the methods are different, they each embody the core principles of agile. Example core principles of agile are empower the team, focus on continuous improvement, work in small batches and deliver increments of value.

The ideas from different methods are often combined by teams to create a custom method for each team.

What is the kanban method?



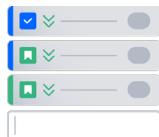
Queue of work items

An agile method used to manage a continuous queue of work items



Improve flow

Remove process bottlenecks to improve "flow" of value



Work in progress

Pull work rather than push (where it makes sense) while limiting work in progress



Visualize and prioritize

Visualize work and Continuously prioritize work items

Kanban is an agile method commonly used to manage a continuous queue of work items. Kanban leverages many of the ideas of the Toyota Production System, which we will discuss later in the course. Some of the ideas used in kanban include:

- Limit work in progress. At any given time, the team should only be working on the amount of work that it can handle sustainably. Limiting work in progress results in small batch sizes.
- Remove bottlenecks to improve flow. Ideally, you would like to have the team outputting a steady stream, or flow of work. The reality is that there will be bottlenecks in the process. Issues will become stuck in certain places and may pile up in other places. These can be because of a problem with the process itself, or just the reality of the complexity of the issues. Either way, the team works together to remove these bottlenecks once they are identified. Attempting to remove bottlenecks forces the team to "see the whole" process rather than just a portion of it. When removing the bottleneck, it is best to find the root cause and fix the process there.
- Pull work rather than push, where it makes sense. For many steps of a process, it is better for someone working on the next step to pull the work from the previous step rather than having it pushed on them when the previous step is finished. We will discuss this idea more a little later.

[https://en.wikipedia.org/wiki/Kanban_\(development\)](https://en.wikipedia.org/wiki/Kanban_(development))

Why choose Kanban?

Lightweight and efficient

Very lightweight and efficient to use.

Evolutionary approach

Evolutionary approach of transforming to agile.

Service-oriented workflows

Works well if the workflow is service-oriented: operations, support, maintenance development, new hire funnel.

Multiple teams and projects

Supports multi-team and multi-project workflows.



Why would a team choose the kanban method?

Kanban is a very lightweight and efficient agile method. All agile methods, by definition, are lightweight, but kanban is lighter than the others, like scrum. You can think of the kanban method as a bare-bones way to achieve agility. This means that it is easy to understand and begin to use. Some teams find that they are more efficient using kanban than other agile methods.

The kanban method can also be an evolutionary approach to agile transformation. Your existing team, in their current roles, can begin to use kanban. Kanban can be done without reorganization, new types of meetings, or new roles. You can start implementing it now. You can then continuously improve to become more agile over time.

The kanban method works well if the workflow is service-oriented. Examples include some of the work of the operations team, support requests, maintenance development, and a human resource department's new hire funnel. Anywhere there is a continuous flow of issues can be a good candidate for kanban. This does not mean that kanban should not be used to develop products. It is used to develop products as well.

Kanban supports multi-team and multi-project workflows. An issue can be moved among teams using a single board or multiple boards specific to each team. Each team can accomplish the issue's work in any way they choose.

Topics

Kanban method overview

Improving flow

Pull vs push



Continuous flow of work items

The screenshot shows a Kanban board interface with the following columns and items:

- BACKLOG 1:** add item 6 (PROJ-6)
- SELECTED FOR DEVELOPMENT 2:** add item 3 (PROJ-3), add item 5 (PROJ-5)
- IN PROGRESS 1:** add item 4 (PROJ-4)
- DONE 2:** add item 1 (PROJ-1)

A large green arrow labeled "flow" points from left to right, indicating the direction of work items moving through the board. A callout box on the left says "continuously add and prioritize new work". A callout box on the right says "continuously finish work". A note at the bottom right says "we're only showing recently modified issues".

Kanban boards focus on improving the flow of issues through the workflow. New issues are continuously added to the backlog and are constantly being prioritized, usually by the business team. Once the issues are ready to be worked on, they are pushed to the “Selected for Development” column and flow through the workflow statuses. Work is continuously being finished before starting new work. When a team member is ready to work on a new issue, they choose the top issue under selected for development and drag, or pull, the issue to the “In Progress” column. This ensures that the team is always working on the most critical issue defined by the team.

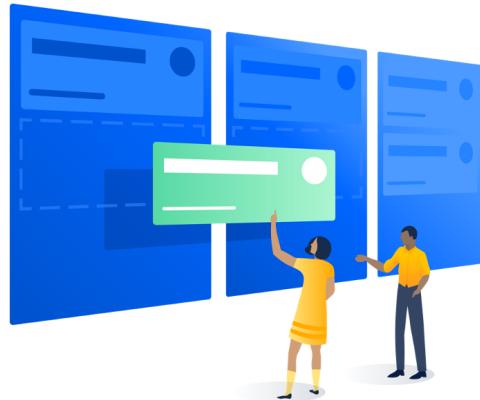
Improving flow- limit work in progress (WIP)

HOW?

Specify the minimum and/or maximum number of issues allowed in certain project board columns.

WHY?

Better flow, limits waste and promotes teamwork.



Limiting work in progress (WIP) is done by specifying the minimum and/or maximum number of issues allowed in specific board columns. Limiting work in progress has many benefits.

If you limit WIP, fewer issues are being worked on at one time, which leads to a better flow of work getting finished. Because of the WIP limits, the team focuses on completing the WIP before starting new issues. This focus results in less multitasking, which is good because multitasking decreases productivity. The delivery of issues is faster with less work in progress because issues are not piling up in specific columns waiting for work to be restarted. Limiting the WIP means that issues will achieve the “done” status. This means that the work is done and can be provided to the customer. Moving issues into the “done” status is the primary measure of the team's progress. By limiting work in progress, any bottlenecks in the process are quickly identified because there are relatively few issues in progress at any time, and problems tend to be easily visible. Identifying and fixing bottlenecks is a way to continuously improve the team's work.

Limiting work in progress also limits the amount of waste in the process. If there is an inventory buildup of issues in a status, there is waste because work is delayed. Limiting WIP also limits the rework that may need to be done if a problem is caused in an early process step but identified later. There are simply fewer issues to fix.

Limiting work in progress has the effect of promoting teamwork. Because the WIP limit restricts the number of issues in a status, the team tends to work together to clear up any blockages.

The link at the bottom of the slide is to a good article on work-in-progress limits in case you are interested in more discussion.

<https://www.atlassian.com/agile/kanban/wip-limits>

Column under minimum limit

The screenshot shows a Jira Kanban board titled "TK board". The board has four columns: "BACKLOG 2", "SELECTED FOR DEVELOPMENT 1", "DONE", and a fourth column on the far right. A red arrow points to the "SELECTED FOR DEVELOPMENT 1" column. The column header has a yellow background and the text "MIN 2" in a small box. There is one card in this column: "As a user, I can set a profile picture" (issue TK-2). The "BACKLOG 2" column has two cards: "As a user, I can cancel a trip" (issue TK-4) and "As a user, I can book a trip" (issue TK-3). The "DONE" column has one card: "As a user, I can create an account" (issue TK-1). At the top of the board, there are various navigation and settings icons.

Here we have configured a minimum WIP limit. You can see that the Selected for Development column has a “Min 2” indication next to the column name, meaning that at least two issues should always be in that column. Since we only have one issue in this column, it is highlighted in yellow. This signal notifies the team that issues should be added to the column. This is a way to ensure that the team always has work items to work on.

(Note that minimum WIP limits are currently not supported on cloud team-managed boards.)

Column over maximum limit

A screenshot of a Jira Kanban board titled "TK board". The board has three columns: "BACKLOG", "SELECTED FOR DEVELOPMENT", and "IN PROGRESS". A red arrow points down to the "IN PROGRESS" column. The "IN PROGRESS" column has a maximum WIP limit of 2, indicated by a red box next to the column name. There are three issues in this column: "As a user, I can book a trip" (status MIN 2), "As a user, I can set a profile picture" (status MIN 2), and "As a user, I can create an account" (status MIN 2). The "DONE" column contains one issue: "Q. See older issues". At the top right of the board, there is a "MAX 2" indicator.



Here we have configured the In Progress column with a maximum WIP limit. The “Max 2” indication is next to the column name. We have three issues with a Review status, so the column has turned red. This signals the team to finish this work before starting more issues.

Notice that we don't need to limit the backlog or done columns because those statuses do not contain the team's work in progress.

This is a way to prevent the team from multitasking and not finishing the work items they start.

What should WIP limits be set to?

- Could start with no WIP limits
- Add WIP limits as the process shows problems
- Could set WIP limits to discourage multitasking
- Could set WIP limits on steps that the team neglects



What should WIP limits be set to? The short answer is that this depends on the specific project and the team. Here are some ideas.

You could start with no WIP limits and see if the issues flow nicely.

You could add limits as the process starts to show problems. For instance, if sometimes there are no items in the “Selected for development” column, the flow is negatively affected because the team can finish an issue and have no issues to work on. You might want to set a minimum limit on the “Selected for development” column to ensure this doesn't happen.

You could also set WIP limits to discourage multitasking. For example, if the team has seven members, you should set limits so each member works on only one issue at a time.

You could also set WIP limits on steps the team tends to neglect. For example, this workflow has a review status after the in-progress status. This step ensures that other team members have looked at the work before it is done. If issues are piling up in the review column, you could set a maximum limit on that column to ensure that the issues flow to the done column.

Topics

Kanban method overview

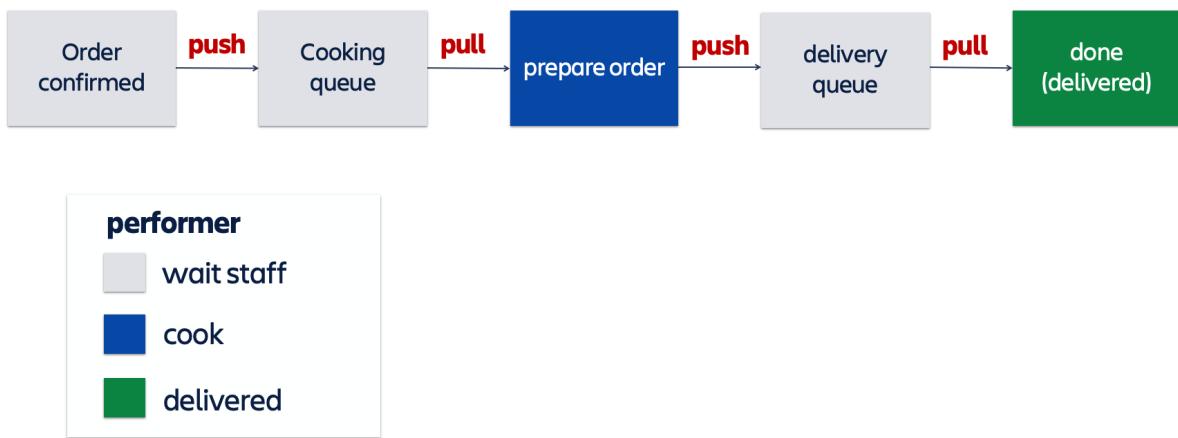
Improving flow

Pull vs. push



Pull vs push in process steps

Performers either push work to the next step or pull from the previous step



The performer of a step in a workflow either pulls the work from the previous step or has the work pushed to them. If we go back to our example workflow where a restaurant takes and delivers orders, we can see that sometimes the work is pushed and sometimes pulled.

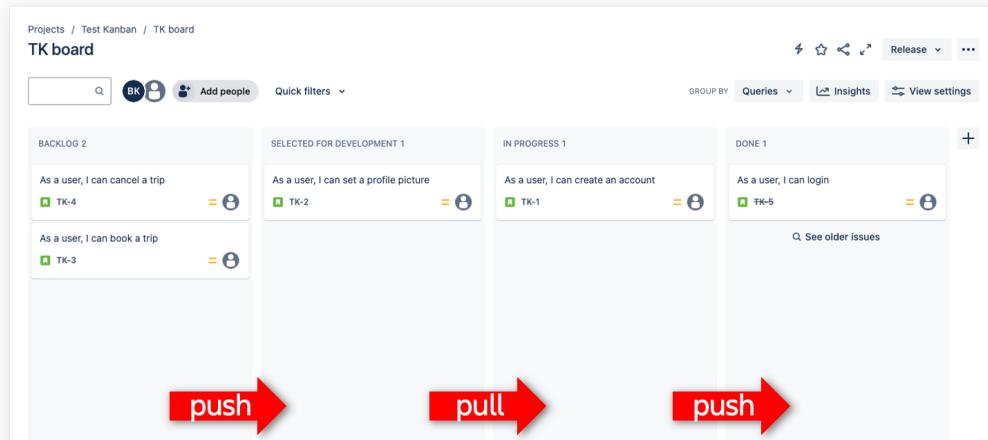
After the wait staff takes the order, they usually push it to a queue for the cook to start when they can. When the cook has the availability to start preparing the order, the cook pulls the order from the queue and places it in the WIPs location.

After preparing the food, the cook pushes the order to the wait staff's delivery queue.

When the wait staff has availability, they pull the order from the delivery queue and deliver it to the customer.

So, you can see that in this workflow, sometimes the performer of the work pulls the work from the previous step, and other times the work is pushed to them from the previous step. Overall, this process is a pull process because the cook only prepares the orders after receiving them. If no customers come in, the cook never prepares any orders. If this were a push process, the cook would prepare the food ahead of time and wait for customers to come in and take the food. If no customers come in, the food could be wasted.

Pull vs push



Pushing and pulling of issues can happen on kanban boards.

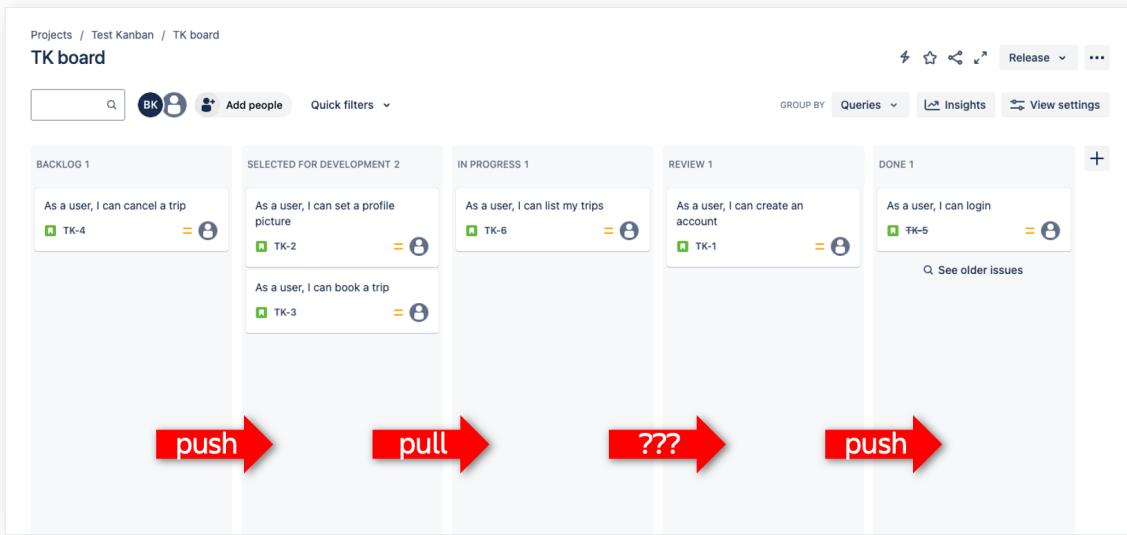
Typically, an issue is pushed onto the “Selected for development” column when it is ready to be worked on.

An issue in the “Selected for development” column is only pulled into the “In progress” column when a person is ready to work on the issue. This is the main reason why the overall process is a pull system.

The people doing the work pull issues only when they have availability.

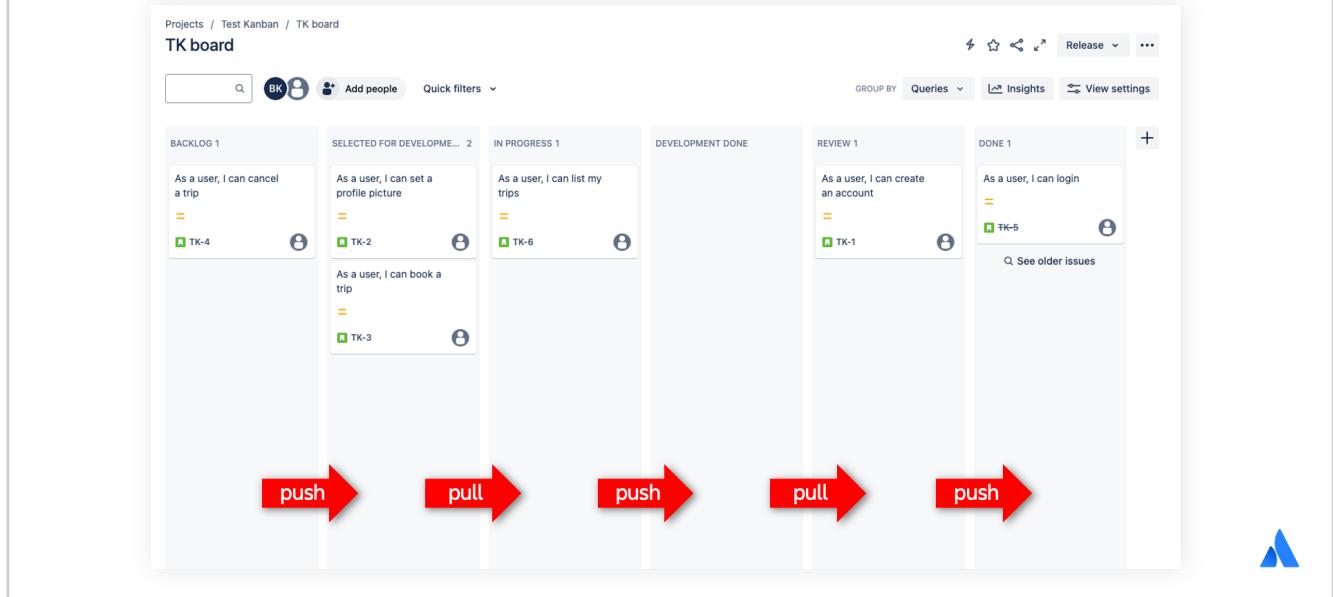
When the work of the issue is complete, the person that did the work pushes the issue onto the done column.

Adding queues to enable pull (1 of 2)



You may want to add queue columns to your workflow. For example, in this workflow, we have a review column before the done column. An issue in the review column usually means the review work is underway. However, when work in the In Progress column is complete, if you push it into the review column, a reviewer may not be available to work on it. If a reviewer pulls the work from the “In progress” column, the work on the issue might not be done. This is a good case for inserting a queue before the review column.

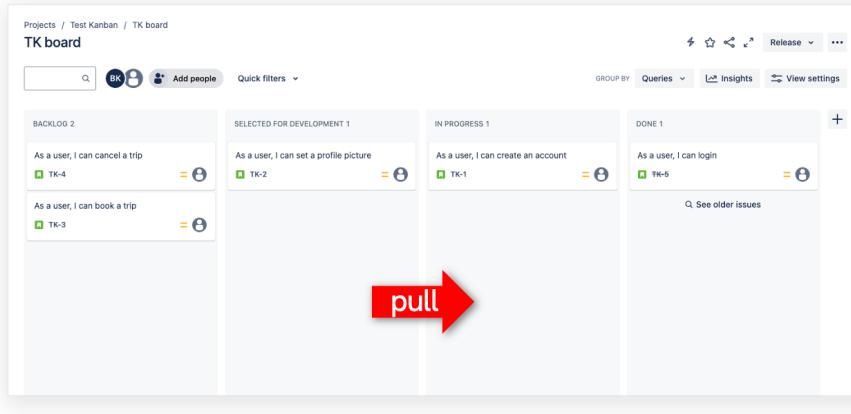
Adding queues to enable pull (2 of 2)



Here we have added a queue called “Development done” to the workflow. When the work of an issue in the “In progress” status is done, that performer can push the issue to the “Development done” queue. When a reviewer is available, they can pull the issue into the “Review” column. Adding this queue makes the exact status of the work item easier to see.

Why pull?

- Empowers the team - team members select work, they are not assigned work
- Maintains a sustainable pace



We have seen that pulling work is common in the kanban method. We will also see later that most agile methods pull work because the development team decides how many issues can be worked on and pulls an issue to start the work.

Pulling work is common on agile teams because it enables team members to select their work. This is what happens in a self-organizing, empowered team. They are not assigned work.

Pulling also maintains a sustainable pace. A performer only pulls more work when they are ready, instead of work being pushed to them that they may or may not be able to keep up with.

Are you getting it?



1. Which of the following statements is true about Kanban method?
 - a. Work is continuously being finished before starting new work.
 - b. Kanban boards focus on improving the flow of issues through the workflow.
 - c. New issues are continuously added to the backlog and are always being prioritized, usually by the business team.
 - d. Once the issues are ready to be worked on, they are pushed to the selected for development column.
 - e. All of the above.



The answer is on the next slide.

Did you get it?



1. Which of the following statements is true about Kanban method?
 - a. Work is continuously being finished before starting new work.
 - b. Kanban boards focus on improving the flow of issues through the workflow.
 - c. New issues are continuously added to the backlog and are always being prioritized, usually by the business team.
 - d. Once the issues are ready to be worked on, they are pushed to the selected for development column.
 - ✓ e. All of the above.



Answer: e

Are you getting it?



2. Which of the following statements about work in progress limits are true?
 - a. They slow down the flow of the process.
 - b. They help the team continuously improve their work.
 - c. They help identify and fix bottlenecks in the process quickly.



The answer is on the next slide.

Did you get it?



2. Which of the following statements about work in progress limits are true?

- a. They slow down the flow of the process.
- b. They help the team continuously improve their work.
- c. They help identify and fix bottlenecks in the process quickly.



Answer: b and c

Are you getting it?



3. An airport deli provides two choices - you can buy a prepared sandwich from the refrigerator or order a sandwich at the counter and have it prepared for you. From the perspective of preparing the sandwich, which uses a pull system?
 - a. Both the refrigerator and the counter.
 - b. The refrigerator only.
 - c. The counter only.



The answer is on the next slide.

Did you get it?



3. An airport deli provides two choices - you can buy a prepared sandwich from the refrigerator or order a sandwich at the counter and have it prepared for you. From the perspective of preparing the sandwich, which uses a pull system?
- a. Both the refrigerator and the counter.
 - b. The refrigerator only.
 - c. The counter only.



Answer: c

Takeaways



- Kanban is a lightweight, agile method
- A board should have a continuous flow of issues moving from backlog to done columns
- Work-in-progress limits can improve the flow of value by focusing the team



Lab 5 – Kanban Method

- Explore WIP limits



6 Scrum Overview I - Artifacts



What will you learn?



- Define scrum
- Describe an increment
- Identify scrum artifacts
- Define velocity



Topics

What is scrum?

Scrum artifacts



“Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems.”

The 2020 Scrum Guide™



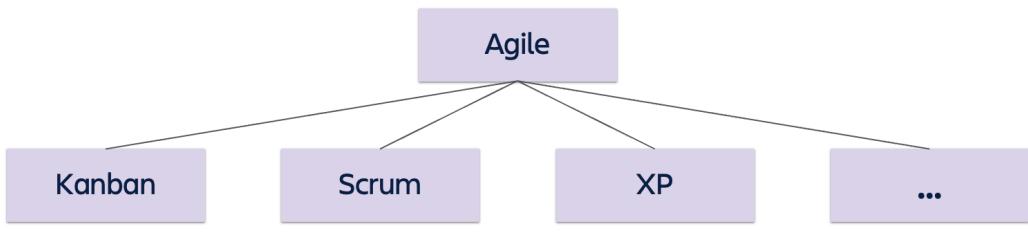
According to the Scrum Guide, written by the creators of scrum, scrum is a lightweight framework that provides adaptive solutions to help teams develop, deliver, and sustain complex products. It is a relatively simple framework for dealing with complex, unpredictable projects.

By framework, we mean that scrum contains basic structure and ideas for completing a project. The Scrum Guide refers to scrum as a process framework for your project management and work techniques rather than a standalone process or definitive method. Every team and every project is different, so the basic ideas of scrum can be customized to suit your specific project. This is contrasted to a rigid methodology in which every project is executed similarly.

Much of what we will discuss here is discussed in the Scrum Guide. It is free, relatively short, and very well written. It is highly recommended that you read it.

What is scrum?

Scrum is a way of achieving agility.

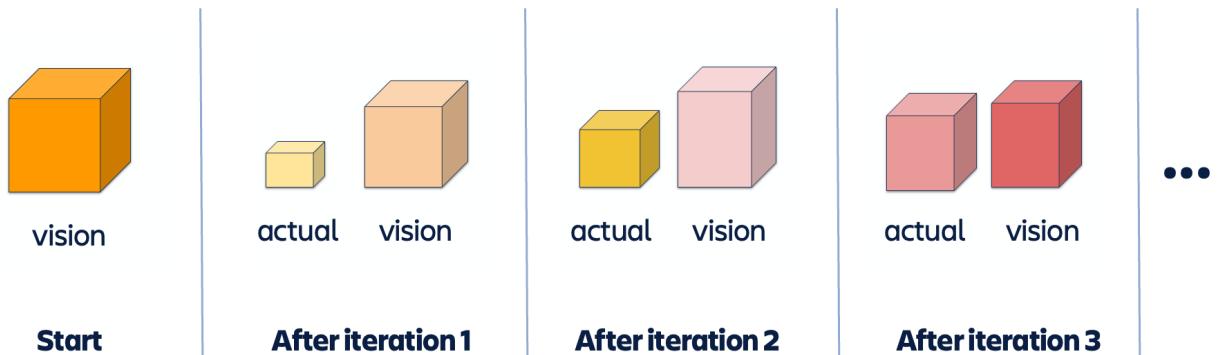


Scrum is a way of achieving the idea of agility.

Previously, we have seen that kanban is another option. You can think of agile as a mindset and the methods such as kanban, scrum, and XP as ways of achieving agility.

<https://www.scrum.org/resources/scrum-guide>

Continuous learning



A key component of scrum and agile in general is continuous learning. Scrum projects start with a vision. This is the initial desired result of the project. The stronger the vision at the start of the project, the better. The product is then worked on an iteration at a time.

After the first iteration, we have some of the product's features. Even though that product only has a few features, it can be considered usable with respect to those features. That product should have value and can be given to the customer. After the first iteration, notice that our vision has slightly changed. We have learned from implementing the first product features and adapted our vision accordingly.

After our second iteration, the product contains the work of the first iteration and the work of the second iteration. It now has more features that the customer will value. Because of continuous feedback, we improved some of the features from the first iteration. Notice that our vision has again been adapted.

After the third iteration, our product is closer to our vision, but our vision has again changed slightly. You can see that we are building the project incrementally because we are building it piece by piece, and we are building it iteratively because the product improves as we learn along the way. That is why scrum is an incremental and iterative approach to building a project. Building and improving product parts continues indefinitely, allowing the product to stay relevant in a changing marketplace.

If the product being built has a physical aspect, such as a rocket, the result of an iteration may be a prototype rather than a part of the completed product.

Increment



Usable product

A usable product that may be given to the customer.

Definition of Done

Meets the organization's "definition of done"

Cumulative

Contains the work of the current iteration, as well as all prior iterations



At the end of every iteration, a product called the increment is ready. Here we see that three iterations have created three increments of the product.

An increment is a usable product that may be given to the customer. The organization always has the option to release the increment.

Each increment must meet the organization's agreed-upon definition of done. This includes quality and security standards and other organizational requirements, such as requiring documentation for each feature.

Each increment contains the work of the current iteration, as well as the work of all prior iterations. In other words, an increment is the state of the product after an iteration.

Providing the customer with the increment will help manage expectations and collect useful feedback.

Sprint

Time-boxed period

A time-boxed period used to work on an increment of the product.



Typical length

A sprint is usually 1-4 weeks. Typically, 2 weeks.

Cumulative

Contains the work of the current iteration, as well as all prior iterations



In scrum, the iterations are called sprints. A sprint is a time-boxed period used to work on an increment. The period of a sprint is fixed. In general, you do not shorten or lengthen the duration of the current sprint. Sprints usually have a duration of one to four weeks, with two-week sprints being typical. Shorter sprints create an opportunity for more adaptation. Longer sprints allow for more work to be done in a single increment. It's up to each team to decide on the appropriate sprint length.

Scrum framework components



Artifacts

- Product backlog
- Sprint backlog
- Increment



Scrum team

- Product owner
- Scrum master
- Development team



Events

- Sprint,
- Sprint planning meeting
- Daily scrum
- Sprint review
- Sprint retrospective



There are three main parts of the scrum framework.

Artifacts are tools that allow for transparency of the project. They allow anyone with access to them to see the current state of the project. We will talk more about each of the artifacts and the commitment teams should follow for each artifact.

The second part of the scrum framework is the scrum team. In the next module, we will discuss the roles of product owner, scrum master and development team members. We will also explain the role of stakeholders in scrum.

The third main part of the scrum framework is the events related to scrum. These are also called ceremonies or meetings. The sprint guide considers the sprint as a container event for the other events. In the next module, we will discuss the sprint planning meeting, daily scrums, the sprint review and the sprint retrospective.

Topics

What is scrum?

Scrum artifacts



Benefits of scrum artifacts



Transparency

It ensures project transparency.



Communication

It enables shared understanding of the work.



Flexibility

It enables inspection and adaptation to change.



The primary purpose of the artifacts is to provide project transparency. Anyone with proper access can use the artifacts to see the current state of the project, including the project's history and plans. This enables the team to have a shared understanding of the project so that everybody is on the same page. The scrum artifacts enable inspection and adaptation inside and outside scrum meetings.

Scrum Artifacts

Product backlog

Sprint backlog

Increment

To do list

An ordered, ever-changing to do list for the project.

Different work items

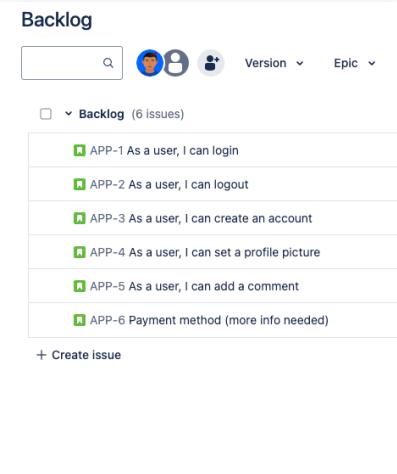
Can include features, improvements, bug fixes, etc.

Prioritized

High priority issues at the top should include more detail.

Refined

Modifying the product backlog is called product backlog refinement.



The screenshot shows the Jira Backlog interface. At the top, there are search and filter buttons, followed by tabs for 'Version' and 'Epic'. Below this is a section titled 'Backlog' with a dropdown menu showing '(6 issues)'. A list of six issues is displayed, each with a small icon and a brief description:

- APP-1 As a user, I can login
- APP-2 As a user, I can logout
- APP-3 As a user, I can create an account
- APP-4 As a user, I can set a profile picture
- APP-5 As a user, I can add a comment
- APP-6 Payment method (more info needed)

At the bottom of the list is a button labeled '+ Create issue'.



A product backlog is an ordered, ever-changing to-do list for the project. It contains issues that still need to be part of any sprint. The Scrum Guide refers to issues as items. You might also hear them referred to as stories. Constant feedback means that the product backlog is constantly changing. Here, you can see the product backlog in Jira. It is located under the Backlog tab for scrum projects. This product backlog contains six issues. When you add issues to a scrum project, they are automatically placed in the product backlog. The product backlog can include issues representing features, improvements, bug fixes, and any other type of issue you would like. The product backlog is ordered. Issues near the top of the backlog are the closest to being worked on, so they usually have more details than the lower items. Modifying the product backlog is called product backlog refinement. You might also hear this referred to as backlog grooming. According to the Scrum Guide, each scrum team decides how to refine the product backlog, but it should consume at most 10 percent of the development team's time.

Scrum Artifacts

Product backlog

Sprint backlog

Increment

Subset of backlog

It's a subset of the product backlog.

Sprint work items

It's the list of issues to be completed in the sprint.

Detailed

It includes plan on how to accomplish the work of the issues.

The screenshot shows a Jira board titled "Backlog". At the top, there is a header with project navigation and search fields. Below the header, there are two main sections: "Sprint 2" and "Backlog".
Sprint 2: Contains three issues:

- APP-1 As a user, I can login
- APP-2 As a user, I can logout
- APP-3 As a user, I can create an account

Backlog: Contains three issues:

- APP-4 As a user, I can set a profile picture
- APP-5 As a user, I can add a comment
- APP-6 Payment method (more info needed)



Here, we have dragged three of the issues from the product backlog to the sprint. The list of issues to be completed during the sprint is called the sprint backlog.

The sprint backlog includes a plan on how to accomplish the work of the issues. In Jira, this usually means that before starting the actual sprint, more details are added to the issues in the sprint backlog. Those details describe how the work of the issues will be done. Some of those details can be added during the sprint, as the team learns more about the issues.

Scrum Artifacts

Product backlog

Sprint backlog

Increment

Deliverable

The deliverable product that was produced by completing the work in the sprint backlog.



Definition of Done

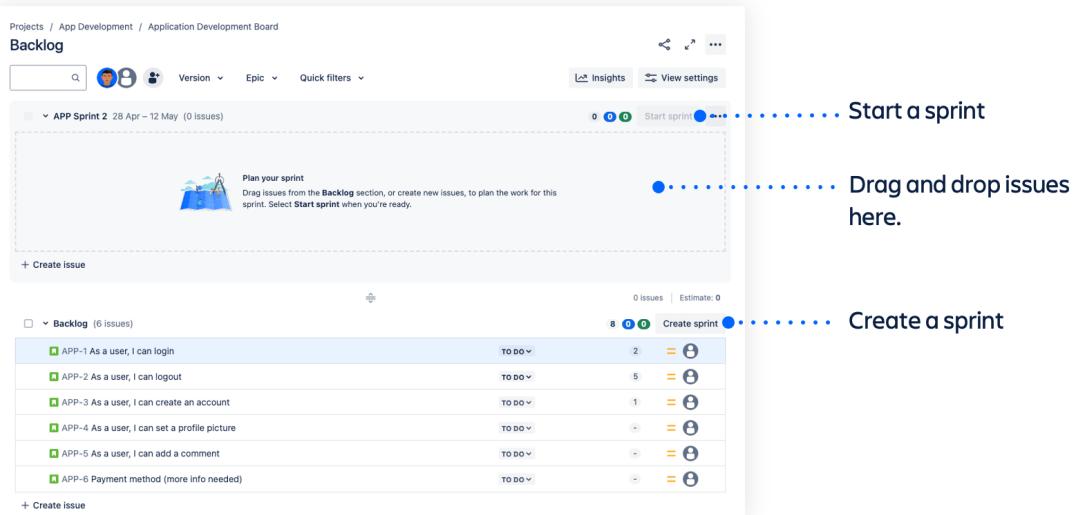
The definition of done describes the state of the increment.



Here, we have dragged three of the issues from the product backlog to the sprint. The list of issues to be completed during the sprint is called the sprint backlog.

The sprint backlog includes a plan on how to accomplish the work of the issues. In Jira, this usually means that before starting the actual sprint, more details are added to the issues in the sprint backlog. Those details describe how the work of the issues will be done. Some of those details can be added during the sprint, as the team learns more about the issues.

Creating and starting a sprint



To create a sprint in Jira, click the **Create sprint** button in the product backlog. After clicking the **Create sprint** button, Jira creates an empty sprint. In this example, Jira named the sprint using the project key, APP, and the sprint number, 2, in our case. You can see that Jira invites you to drag issues from your product backlog into the sprint.

Estimation – story points

- Relative measure of work (effort) required to complete the story
- Used to help decide how many stories can be completed in the sprint

The screenshot shows the Jira Application Development Board. At the top, there is a header with project navigation, user icons, and search/filter options. Below the header, the main interface is divided into two main sections: the Sprint Backlog and the Product Backlog.

Sprint Backlog: This section is titled "APP Sprint 2 28 Apr – 12 May (3 issues)". It contains three items, each represented by a card:

- APP-1 As a user, I can login (Story Points: 2)
- APP-2 As a user, I can logout (Story Points: 5)
- APP-3 As a user, I can create an account (Story Points: 10)

Below the Sprint Backlog, there is a summary: "3 issues | Estimate: 17".

Product Backlog: This section is titled "Backlog (3 issues)". It contains three items:

- APP-4 As a user, I can set a profile picture (Story Points: 3)
- APP-5 As a user, I can add a comment (Story Points: 1)
- APP-6 Payment method (more info needed) (Story Points: 1)

Below the Product Backlog, there is a summary: "3 issues | Estimate: 17".



As part of planning for sprints, estimating how much work (or effort) an issue will take is common. Story points are the most common estimation statistic. In Jira, you can use story points, hours, issue count, or create your own estimation statistic. Story points are a relative measure of the work required to complete an issue. They are not a measure of time. For example, an issue assigned two story points is assumed to involve about twice as much effort as an issue given one story point. You can see that in Jira, there is a field on each issue named Story Points. Here, we have given one story point to this issue. In the sprint backlog, you can see that the story points are shown in the gray bubbles, along with a total estimate of 17 points for the sprint backlog. You can see that the issue remaining in the product backlog has been assigned three story points. Story points help the team decide how many issues can be completed in a sprint.

Sprint details

Start sprint

2 issues will be included in this sprint.

Sprint name: *
PRJ Sprint 1

Duration: *
2 weeks

Start date: *
01/Jul/20 4:38 PM 

End date: *
15/Jul/20 04:38 PM 

Sprint goal:



After clicking the **Start sprint** button, the Start sprint screen appears. Jira starts by reminding you that you have added two issues to the sprint backlog. You can modify the sprint name, specify the sprint's duration, and specify the start date for the sprint. You can see that you can have the sprint started at a later date, so you don't have to click the start sprint button on the first morning of the sprint. You could also set up multiple sprints at one time.

Sprint goal

- Represents the objective of the sprint's increment
- Is reached by completing the sprint backlog
- Does not change during the sprint
- The sprint is a success if the sprint goal is reached

Start Sprint

3 issues will be included in this sprint.

Required fields are marked with an asterisk *

Sprint name *

Duration *

Start date *

Planned start date: 4/28/2024, 10:51 PM
A sprint's start date impacts velocity and scope in reports. [Learn more](#).

End date *

Sprint goal

[Cancel](#) [Start](#)



The start sprint window also contains a place to enter the sprint goal. The sprint goal represents the objective of the sprint's increment, and the team agrees to the goal. The sprint goal is reached by completing the issues in the sprint backlog. A scrum rule is that the sprint goal does not change during the sprint. The sprint is considered a success if the sprint goal is reached.

Why have a sprint goal?

1. Provides coherence to the product increment
2. Enables flexibility with the sprint backlog
3. Helps team stay committed to the end goal.

Start Sprint

3 issues will be included in this sprint.
Required fields are marked with an asterisk *

Sprint name *

Duration *

custom

Start date *

4/15/2024 1:05 PM

Planned start date: 4/28/2024, 10:51 PM
A sprint's start date impacts velocity and scope in reports. [Learn more](#).

End date *

5/12/2024 10:51 PM

Sprint goal

Cancel Start



There are two significant reasons to have a sprint goal.

1. It provides coherence to the product increment. The features are related, so the product increment is valuable rather than building a collection of unrelated features. This also results in the scrum team working together to achieve the sprint goal.
2. It enables flexibility with the sprint backlog. Projects are complex, and even though the sprint duration is relatively short, the team can not predict the future and will learn and adjust during the sprint. There has to be flexibility somewhere. The sprint goal remains fixed during the sprint, but the issues that achieve the sprint goal can be modified as long as quality is not decreased. This means there is flexibility in the makeup of the sprint backlog as the sprint is worked on. The sprint goal guides decisions as the team makes adjustments.
3. It helps the development team to stay committed to the goal of the sprint.

Sprint board

Only contains issues from the sprint backlog

The screenshot shows a Jira Sprint Board for 'PRJCTB Sprint 2'. The board has three columns: 'TO DO', 'IN PROGRESS', and 'DONE'. An issue card titled 'SSP-10' is in the 'IN PROGRESS' column. A tooltip for this card says: 'Update task status by dragging and dropping from column to column >> Try dragging this task to 'Done''. Below the board, there are sections for 'Other Issues' and 'Instructions for deleting this sample board and project'. A sidebar on the left shows navigation links like 'ProjectB1', 'Planning', 'Development', and 'Project settings'.

A sprint has a sprint board. Notice it only contains the issues in the sprint backlog. Issues in the product backlog or those assigned to other sprints are not shown on the sprint board.

Scrum reports - burndown chart



A burndown chart shows the progress that the team makes during a sprint.

1. The sprint backlog starts with several issues, each with an associated number of story points or another estimation statistic. The total number of starting story points is shown on the left of the chart. This is the number of story points that the development team estimated that it would complete in the sprint. In our case, this sprint has 3 story points.
2. The gray guideline indicates the number of story points that should remain on a given day, assuming a linear burndown of story points. On the last day of the sprint, the guideline reaches zero story points. This means the sprint's work should be finished on the last day.
3. Notice that the chart shows the non-working days, and the guidelines assume no progress will be made.
4. As the sprint is underway, Jira will automatically update the burndown chart as the status of the issues is updated by the team. The red line shows the actual number of remaining story points over time.
5. On Apr 27, one story point was completed.
6. On the last day of the sprint, the remaining 2 story points were completed. Consulting this chart is an easy way to see if the team is on track for the current sprint.
7. If the red line is below the gray line, your team is on track to complete all the story points and reach the sprint goal.
8. If not, the team may need to make adjustments to achieve the sprint goal.

The link shown contains more information on burndown charts.

<https://www.atlassian.com/agile/tutorials/burndown-charts>

Velocity

Rate at which the team accomplishes work

- Usually, it's the number of story points completed per sprint



Velocity represents the rate at which the team accomplishes work.

Usually, it is the number of story points completed per sprint. Some teams use an estimation statistic other than story points, so in that case, velocity measures some other units completed per sprint.

You can see the team's velocity of a single sprint by looking at a burndown chart. The team completed three story points in this sprint, so its velocity is three.

Are you getting it?



1. Which one of the following statements is true?
 - a. The product vision remains constant in scrum as the team builds the product.
 - b. Scrum is a framework for achieving agility.
 - c. Scrum projects all follow the same practices.



The answer is on the next slide.

Did you get it?



1. Which one of the following statements is true?
 - a. The product vision remains constant in scrum as the team builds the product.
 - ✓ b. Scrum is a framework for achieving agility.
 - c. Scrum projects all follow the same practices.



Answer: b

Are you getting it?



2. Which one of the following statements is true?
 - a. A product backlog is the same thing as a sprint backlog.
 - b. A sprint board includes the product backlog.
 - c. A burndown chart should reach zero by the end of the sprint.



The answer is on the next slide.

Did you get it?



2. Which one of the following statements is true?
 - a. A product backlog is the same thing as a sprint backlog.
 - b. A sprint board includes the product backlog.
 - ✓ c. A burndown chart should reach zero by the end of the sprint.



Answer: c

Are you getting it?



3. What does a sprint backlog contain?
 - a. A list of issues to be completed after the current sprint.
 - b. The list of issues to be completed in the sprint and a plan on how to complete them.
 - c. A prioritized list of all of the issues of the project.



The answer is on the next slide.

Did you get it?



3. What does a sprint backlog contain?
 - a. A list of issues to be completed after the current sprint.
 - ✓ b. The list of issues to be completed in the sprint and a plan on how to complete them.
 - c. A prioritized list of all of the issues of the project.



Answer: b

Takeaways



- Scrum is an agile framework
- An increment is a potentially shippable portion of the project that meets the "definition of done"
- A sprint is a time-boxed period in which an increment is created
- Scrum artifacts provide project transparency, enable shared understanding, and enable inspection and adaptation
- Artifacts include the product backlog, the sprint backlog, the sprint goal, sprint boards and reports
- Velocity is the rate at which the team accomplishes work, usually in story points per sprint



Lab 6 – Scrum overview I

- Explore a scrum project



7

Scrum Overview II – Roles and Events



What will you learn?



- Describe scrum roles
- Differentiate the product owner and scrum master
- Identify common characteristics of scrum events
- Identify the purpose of the sprint planning meeting, daily standup, sprint review, and sprint retrospective



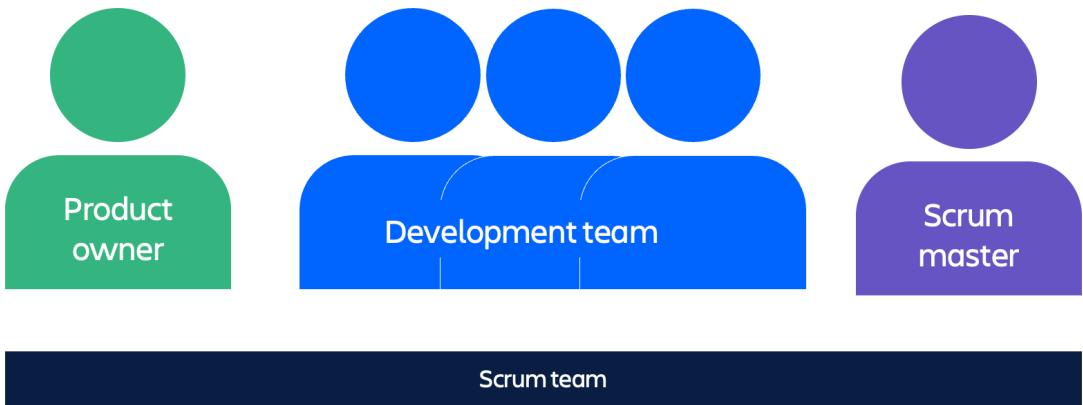
Topics

Scrum roles

Scrum events



Scrum team



A scrum team is made up of three roles: The product owner, the scrum master and the development team.



Values of a scrum team



Cross-functional

Team members have the necessary skills to implement the work in each sprint.



Accountable

Scrum teams are responsible for their work and are accountable for creating a valuable increment every sprint.



Self-organizing

Scrum teams manage their own work and internally decide who does what, when and how.



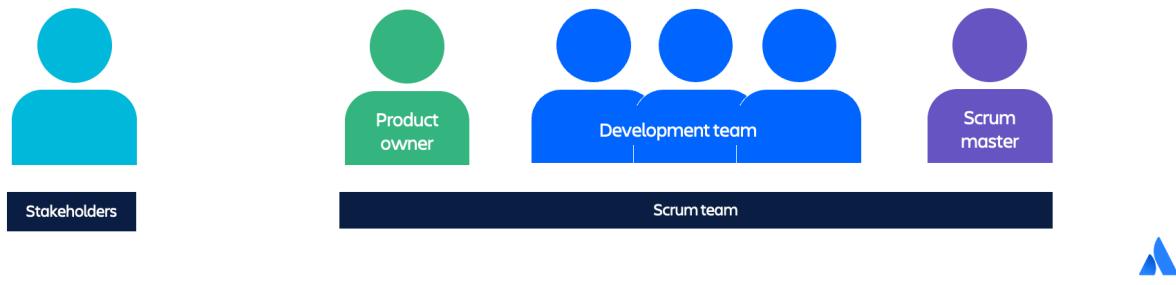
The scrum team is by definition cross-functional, which means, team members can complete the work in a sprint within the team itself. They don't require skill from outside the team to achieve the work of a sprint.

The scrum team is also accountable for producing value at the end of every sprint.

It is self-organizing. The team is responsible for deciding how to organize and do its work.

Stakeholders

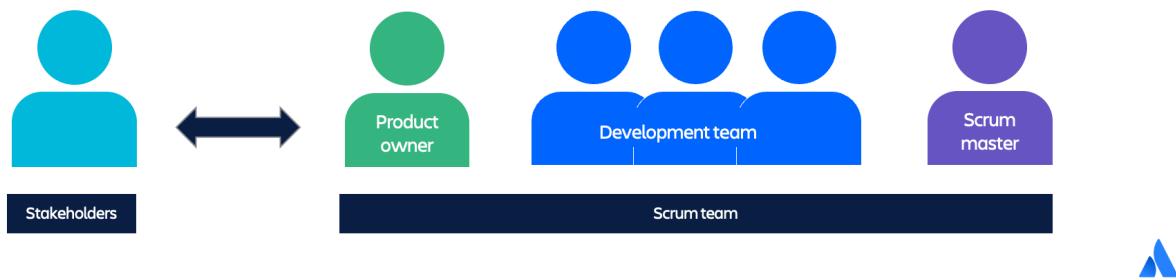
- Others interested in the success of the project.
- Internal- company managers, executives, other scrum teams.
- External- customers, partners, investors.



Stakeholders are not members of the scrum team but are interested in the project's success. Some internal stakeholders, such as company managers, executives, or other scrum teams, rely on the scrum team's work. There are also stakeholders external to the organization, such as customers, partners, and investors.

Product owner

- Responsible for:
 - communicating the product vision
 - maximizing the value of each increment
 - the product backlog
- Interacts with, represents and is accountable to stakeholders



The product owner is a member of the scrum team who is responsible for communicating the product vision. The stakeholders and the scrum team must understand the product vision to work effectively. The product owner is also responsible for maximizing the value of each increment. Each feature that the development team works on should be of high value. <click>The product owner is responsible for the product backlog. Others may help with the product backlog, but the product owner is responsible. Stakeholders primarily interact with the product owner. The product owner represents the stakeholders when they are not part of the discussion, such as during scrum team meetings. The product owner is accountable to the stakeholders for the project's success.

Scrum master

Responsibilities

- promoting and supporting scrum
- improving the day-to-day effectiveness of the team
- protecting the focus of the team
- increasing the transparency of the project

Tasks

- coaching the scrum team and stakeholders on scrum
- removing blocking issues
- facilitating scrum events
- configuring scrum artifacts
- monitoring sprint progress



The scrum master is the scrum team member who is primarily responsible for promoting and supporting scrum for scrum team members as well as stakeholders. It is up to the scrum master to ensure that everyone understands how and why things are done a certain way.

The scrum master is also responsible for the day-to-day effectiveness of the scrum team. This includes ensuring that the team is reaching its goals and continuously improving.

The scrum master is responsible for protecting the focus of the team. This may mean helping to remove bottlenecks or ensuring that those outside interact with the team in helpful ways. In general, scrum masters do what they must to allow team members to focus on their work.

The scrum master is also responsible for increasing the transparency of the project. There should be no surprises about the current status of the project.

The scrum master's tasks vary by project and from day to day. Typical tasks include:

- Coaching the scrum team and stakeholders on scrum and agile
- Helping to remove blocking issues
- Facilitating scrum events
- Configuring scrum artifacts
- Monitoring sprint progress

Product owner vs scrum master



Why separate roles?

- Divide and concur
- Checks and balances



The distinction between the product owner and scrum master roles should now be straightforward. The product owner is responsible for the value of the product. The scrum master is primarily responsible for the effectiveness of the team.

You can think of this as both a divide and conquer approach because the work of the roles is so different and as a checks and balances approach because combining the roles could put too much weight on one of the responsibilities. The separate roles lead to greater team success and sustainability. Combining these roles into one person is discouraged because the roles have different, sometimes competing, primary responsibilities. Combining the roles defeats the checks and balances.

Some companies choose to use a single scrum master for several teams. Some teams have a scrum master who is also a part-time development team member.

Development team

- Cross-functional, adaptive team that does the work of the project
- Responsibilities:
 - estimating issues
 - deciding how much work can be done in a sprint
 - deciding how to organize to do the work of the sprint
 - creating the increment of each sprint
 - ability to modify the sprint backlog during the sprint
- The Scrum Guide recommends three to nine members



The development team is a cross-functional, adaptive team that does the project's work.

Responsibilities of the development team include:

- Estimating issues. This is usually done using story points, but other estimation methods can be used.
- Deciding how much work can be done in the sprint. Only the development team can choose how much work to take on. The development team is assumed to be in the best position to forecast how much work the issues take.
- Deciding how to organize to do the work of the team. This is because the team is an empowered, self-organizing team.
- Creating the increment of each sprint.
- Development team members are the only members allowed to modify the sprint backlog during the sprint. This is sometimes necessary as the team learns and adjusts as it builds.

The Scrum Guide recommends having from three to nine development team members. A three-member team decreases the productivity and quality created by a cross-functional group working together. More than nine members tend to increase the amount of coordination required. Jeff Bezos at Amazon refers to two pizza teams, meaning two pizzas should be able to feed the entire team. It is usually better to create multiple teams to scale scrum to more than nine people.

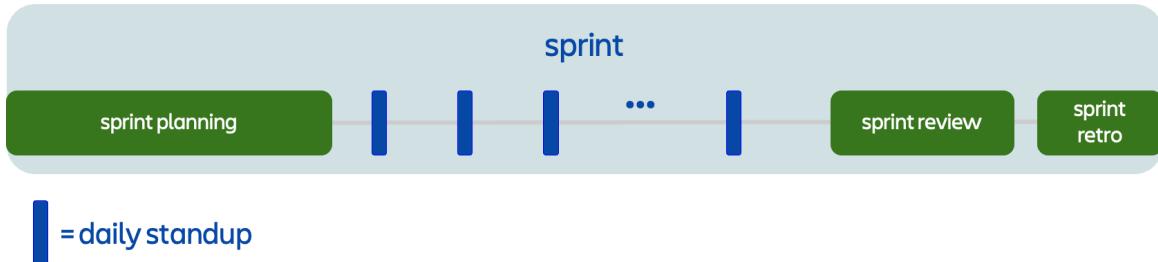
Topics

Scrum roles

Scrum events



Scrum events



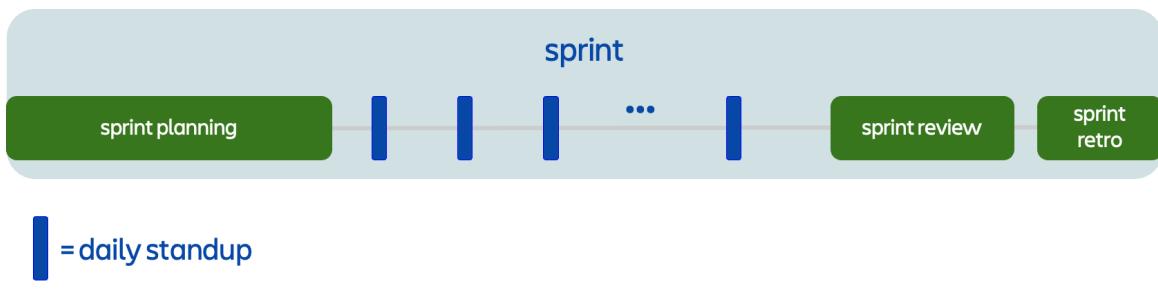
A sprint contains four types of events. You may also hear events referred to as ceremonies or simply meetings. The possibilities are the sprint planning meeting, the daily standups, the sprint review, and the sprint retrospective. Scrum events occur at regular intervals and minimize the need for other meetings. These events are designed to maximize the opportunities for feedback and continuous learning and are vital to achieving agility. In between these meetings is where the work on the sprint issues is completed.

We will discuss each of these meetings separately.

<https://www.atlassian.com/agile/scrum/ceremonies>

Characteristics of all scrum meetings

- Fixed maximum time limit, no minimum time limit
- Meetings are primarily to plan, inspect and adapt
- Primarily about collaborating, not updating status
- Primarily spend time on things of value to all participants



All of the scrum meetings have some common characteristics.

The meetings have a fixed maximum time limit and no minimum time limit. Meetings can never go over their allotted time, but they can be ended early if the purpose of the meeting is achieved.

The meetings are primarily to plan, inspect, and adapt. In agile projects, the planning is distributed throughout rather than the primarily upfront planning of waterfall projects. These meetings are an essential part of that distributed planning. The meetings are used to inspect the project and adapt what the team does based on that inspection. This is a vital part of increasing transparency and continuous improvement.

The meetings are primarily about the team collaborating, not about updating status. The work is visualized, so you don't need a meeting to see the status.

Every team member is responsible for ensuring that the meetings have value and to help modify them to increase their value if necessary. If the discussion moves in a direction that becomes important to only a portion of the participants, it should be moved to a later time outside the meeting.

Sprint planning meeting

- **Attendees:** Entire scrum team
- **Duration:** Typically four hours for a two week sprint
- **Purpose:** Plan the work of the sprint
- **Output:** Sprint goal, sprint backlog



The sprint planning meeting is held at the start of the sprint.

The entire scrum team attends the meeting.

The duration of the meeting is typically four hours for a two-week sprint. If your sprints are four weeks long, this meeting should be twice as long.

The purpose of the meeting is to plan the work of the sprint.

The meeting output is an agreed-upon sprint goal and a sprint backlog.

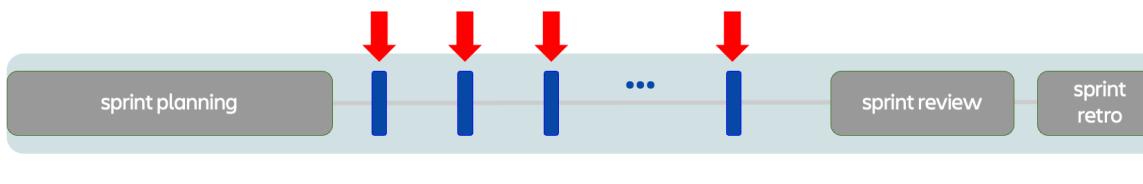
Before the meeting, the product owner usually has a proposed sprint goal and the minimum set of issues that accomplish the goal. These preliminary items often come from the product backlog updated during the previous sprint's review and sprint retrospective meetings. We will discuss those meetings a bit later.

During the meeting, the team usually discusses the sprint goal, modifies the sprint backlog, places story point estimates on issues, and adds details to the issues to better describe the specific work to be done. The development team is responsible for estimating the story points for the work and deciding on how much work can be done during the sprint. They need to plan enough to have an accurate forecast for the amount of work they will agree to. The development team also creates subtasks for the first few days of the sprint. The subtasks are often a day or less of work. This is an example of an empowered team rather than a command and control-based team. The purpose of the meeting has been met when there is an agreed-upon sprint goal and a sprint backlog. The sprint backlog contains the issues that the development team has agreed to complete during the sprint and a plan for completing that work.

Daily scrum

A focused, standing, quick meeting of a very short duration

- **Attendees:** development team and scrum master (primarily)
- **Duration:** 15 minutes
- **Purpose:**
 - Inspect recent progress toward sprint goal
 - Plan the day's work
 - Identify impediments and plans to resolve them
- **Output:** plan for the day



The daily scrum meeting is also called the daily standup. It is a planning meeting that occurs every day, and the participants usually stand as a reminder that it is a short meeting. The meeting usually takes place in the same location at the same time every day.

The development team and scrum master are the primary attendee for the daily standup. Others may attend the meetings, but are usually asked to listen only.

The daily standup is often facilitated by the scrum master. Some teams choose to rotate the facilitator among the team members.

The purpose of the meeting is to inspect recent progress toward the sprint goal, plan the day's work and identify any impediments to the team. The team usually makes plans to resolve the impediments, but the discussion often moves to a discussion that follows the meeting.

The output of the meeting is the plan for the day.

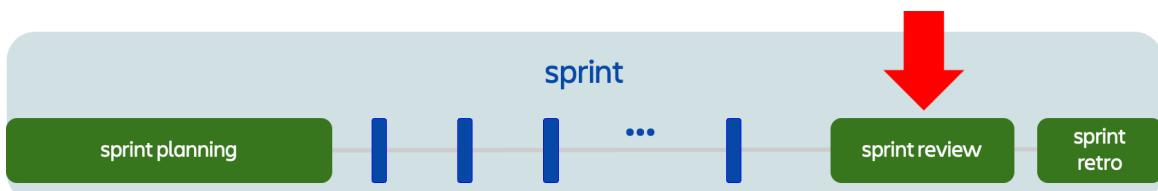
It's important that the daily standup is a collaborative meeting and not simply a status update. The team collaborates to make the best decisions for the day given the latest information. They usually will decide on who will work on specific issues. The plan slightly changes after every daily standup. This is continuous improvement.

The link at the bottom is a link from the Atlassian team playbook on standups. You can find more information there.

<https://www.atlassian.com/team-playbook/plays/standups>

Sprint review

- **Attendees:** scrum team and stakeholders
- **Duration:** typically 2 hours for a 2 week sprint
- **Purpose:** Inspect the increment and collaboratively update the product backlog
- **Output:** first-pass next sprint backlog



The sprint review meeting occurs near the end of the sprint.

It is an informal meeting that includes the scrum team and interested stakeholders.

It is typically a two-hour meeting for two-week sprints.

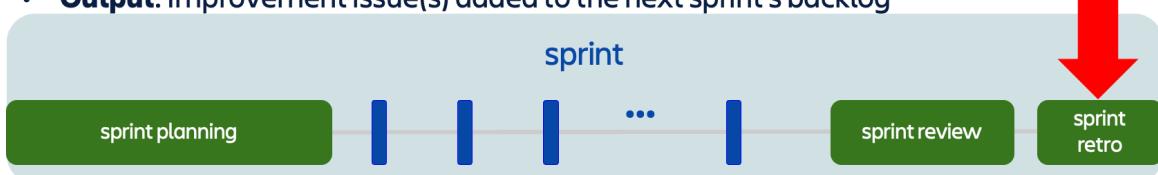
The purpose of the meeting is to inspect the increment created during the sprint and update the product backlog collaboratively. This is a meeting with a lot of feedback on the project and includes a brainstorming session to help decide what to do next.

The meeting output is a first pass at the next sprint's backlog. By the time the sprint planning meeting happens for the following sprint, the team already has a good idea of what they will be working on.

<https://www.atlassian.com/agile/scrum/sprint-reviews>

Sprint retrospective

- **Attendees:** scrum team
- **Duration:** typically 90 minutes for a 2-week sprint
- **Purpose:** team inspects itself, including its processes, tools, and team interaction
- **Output:** Improvement issue(s) added to the next sprint's backlog



The sprint retrospective is the last event of the sprint.

The scrum team attends the retrospective. The scrum master often facilitates this meeting.

A retrospective typically takes 90 minutes for a two-week sprint.

The purpose of the meeting is for the team to inspect itself, including its processes, tools, and team interaction. The retrospective is a positive meeting containing constructive feedback. Everyone should always remember that they are a part of a team. The scrum master usually helps make sure that this is a positive meeting. A retrospective can have a celebratory feel because the team should feel good about having delivered the increment. In a retrospective, the team usually discusses what they should keep doing, what they should stop doing, and what they should start doing. The meeting is about continuously improving the team.

The output of the meeting is to add one or more improvement-related issues to the next sprint's backlog. The team must spend some time on these issues rather than exclusively building the product. The link at the bottom is from the Atlassian team playbook on retrospectives. You can find more information there.

<https://www.atlassian.com/team-playbook/plays/retrospective>

Scrum meetings - summary

| | Sprint Planning | Daily Standup | Sprint Review | Retrospective |
|------------------|-----------------------------|---------------------------------|---|-----------------------------|
| Attendees | Scrum team | Development team (primarily) | Scrum team and stakeholders | Scrum team |
| Duration* | 2 hours | 15 minutes | 2 hours | 90 minutes |
| Purpose | Plan the work of the sprint | Inspect recent work, plan today | Inspect increment, brainstorm next sprint | Inspect team |
| Output | Sprint goal, Sprint backlog | Today's plan | Proposed next sprint backlog | Amended next sprint backlog |

*typical max duration assuming a two week sprint



Here is a summary of the four meetings related to a sprint. It's essential always to ensure that the meetings have high value, and the team should focus on continuously improving their meetings. Notice that the next sprint goal and sprint backlog start to form in the sprint review. The retrospective usually adds one or more issues. This becomes the starting sprint backlog for the sprint planning meeting.

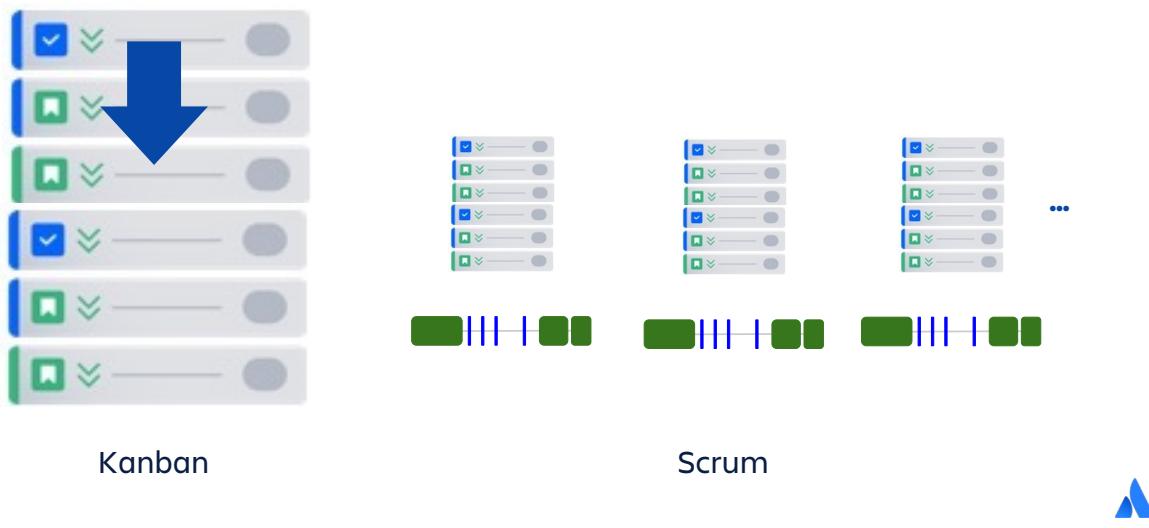
Product backlog refinement

- **Attendees:** product owner and some/all of team
- **Duration:** on going activity
- **Purpose:** create a prioritized backlog with stories ready to be completed
- **Output:** product backlog
 - contains appropriate items
 - prioritized



Product backlog refinement is not performed on the sprint backlog. The sprint backlog is already set, and the team selects stories from the sprint backlog during the sprint. The product backlog contains stories that haven't yet been assigned to a sprint. For the product backlog to be of use, it must contain stories appropriate for the work that needs to be accomplished. The product backlog must be prioritized where the stories at the top are of the highest priority. The product owner and members of the team perform this refinement. Stories in the backlog that are no longer relevant are removed. New stories are created due to new needs. Stories are estimated. Any missing details are added to the stories. By performing product backlog refinement on a regular basis, the product backlog is continually populated with stories ready to be moved into a sprint and worked on by the team. Product backlog refinement isn't a traditional ceremony like standup or any of the others previously discussed. This can be performed during a scheduled meeting or on an impromptu basis.

Kanban vs scrum



Kanban is a smaller framework than scrum. Kanban does not define as much of the agile process as scrum does. For example, it is up to the kanban team to decide which ceremonies/events to use (if any). With kanban, a continuously prioritized backlog of work is presented to the team. When team members are ready for more work, they pull a work item. A kanban project board contains an ongoing queue of items.

With scrum, the team's work is divided into batches of work items completed during sprints. This is a compromise between working with large batches and a continuous queue. A sprint contains four types of meetings. It also includes artifacts such as a sprint backlog, sprint goal, and sprint board. A sprint board only consists of the work items to be completed during the sprint. Like with a kanban project, when a team member is ready for more work, they pull a work item.

It is up to the team to decide whether to use kanban or scrum, some combination of them, or another agile framework/method. If the nature of the work is more of a queue, like with support requests, kanban might be a better choice. Scrum might be a better choice for teams building a product, though some teams could be more effective using kanban. With kanban, the team must decide when and why to meet.

Are you getting it?



1. Which scrum role is responsible for the product backlog?
 - a. Scrum master
 - b. Product owner



The answer is on the next slide.

Did you get it?



1. Which scrum role is responsible for the product backlog?
 - a. Scrum master
 - b. Product owner



Answer: b

Are you getting it?



2. Which scrum meeting is most likely to include stakeholders?
 - a. Sprint planning meeting
 - b. Sprint retrospective
 - c. Daily standup/scrum
 - d. Sprint review



The answer is on the next slide.

Did you get it?



2. Which scrum meeting is most likely to include stakeholders?
 - a. Sprint planning meeting.
 - b. Sprint retrospective.
 - c. Daily standup/scrum.
 - ✓ d. Sprint review.



Answer: d

Are you getting it?



3. Which one of these best describes the purpose of a sprint retrospective?
 - a. Plan the work of the sprint.
 - b. The team inspects itself, including its processes, tools, and team interaction.
 - c. Inspect the increment and collaboratively update the product backlog.



The answer is on the next slide.

Did you get it?



3. Which one of these best describes the purpose of a sprint retrospective?
- a. Plan the work of the sprint.
 - ✓ b. The team inspects itself, including its processes, tools and team interaction.
 - c. Inspect the increment and collaboratively update the product backlog.



Answer: b

Are you getting it?



4. Which scrum role is most responsible for protecting the focus of the team?
 - a. Scrum master
 - b. Development team members
 - c. Product owner



The answer is on the next slide.

Did you get it?



4. Which scrum role is most responsible for protecting the focus of the team?

- ✓ a. Scrum master
- b. Development team members
- c. Product owner



Answer: a

Takeaways



- Scrum roles include:
 - product owner
 - scrum master
 - development team members
 - stakeholders
- Scrum meetings include:
 - sprint planning meeting
 - daily standups
 - sprint review
 - sprint retrospective



Lab 7 – Scrum Overview II

- Create issues in the product backlog
- Create and plan a sprint
- Execute a sprint
- Complete a sprint



8

Quick Search and Basic Search



What will you learn?

- Identify the ways to search in Jira
- Use quick search
- Use basic search



Topics

[Searching overview](#)

[Quick search](#)

[Basic search](#)



Viewing a project's progress



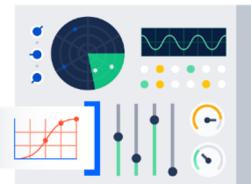
Boards



Search



Reports



Dashboards



Project boards, searching, reports and dashboards help your team view the current and historic status of the project, as well as help plan the project. Using these tools, you can answer most questions that are relevant to the team, either in an ongoing manner or with specific just-in-time team questions. We will cover searching now, but the same techniques are used behind the scenes for reports and dashboards.

Why is searching important?

Adapt your Jira experience to your team's processes

Based on a search

The screenshot shows a Jira Kanban board titled "TK board". The board is organized into six columns: BACKLOG, SELECTED FOR DEV., IN PROGRESS, DEVELOPMENT DONE, REVIEW, and DONE. Each column contains cards representing user stories or tasks. The "BACKLOG" column has three cards: "As a user, I can cancel a trip" (TK-4), "As a user, I can book a trip" (TK-3), and "As a user, I can set a profile picture" (TK-2). The "SELECTED FOR DEV." column has one card: "As a user, I can list my trips" (TK-6). The "IN PROGRESS" column has one card: "As a user, I can create an account" (TK-1). The "DONE" column has one card: "As a user, I can login" (TK-5). The top navigation bar includes links for "Projects / Test Kanban / TK board", "Quick filters", "GROUP BY Queries", "Insights", and "View settings".



Why are we learning about searching? Because it is used to help adapt your Jira experience to your team's processes. We have seen that each agile team is unique and constantly improving, so the tools used must be flexible enough to adapt to changing circumstances.

Boards are an example of a tool used to support agile teams and that is based on a search filter.

Searching

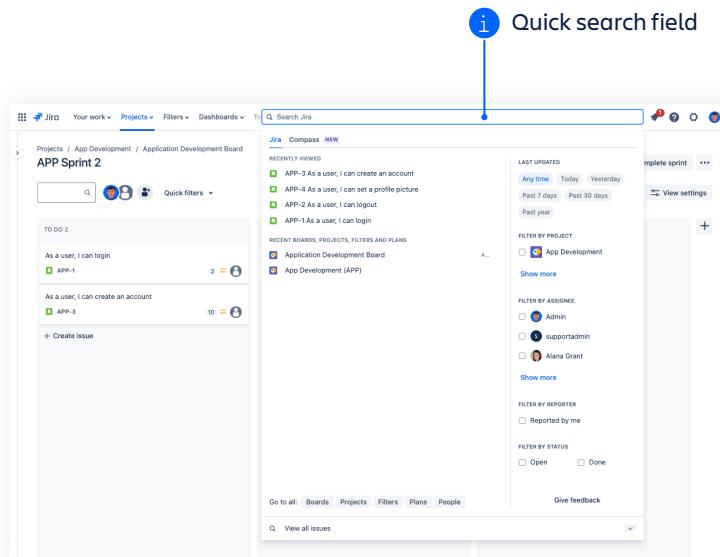
Quick search

Basic search

Advanced search

Filters

Quick filters



There are several topics related to searching in Jira. We will discuss them quickly now and then in more detail in the next module.

The first way of searching is the quick search. This allows you to quickly search for text in issues, board names, project names, filter names and people names using a text-based search. You can also quickly access recently viewed items, as shown here.

Searching

Quick search

Basic search

Advanced search

Filters

Quick filters

The screenshot shows the Jira Issues page for projectA. At the top, there are 'Basic search fields' (Search issues, Project dropdown set to 'Project: projectA', and several filter buttons like Type, Status, Assignee, More, Reset, Save filter) and a 'Basic/advanced search switch' (a button labeled 'BASIC' with a blue info icon). Below the search bar is a table with three rows of issue data:

| Type | Key | Summary | Assignee | Reporter | P | Status |
|------|--------|------------|-------------|----------|---|---------|
| BUG | PROJ-3 | add item 3 | Unassigned | admin | = | BACKLOG |
| BUG | PROJ-2 | add item 2 | Unassigned | admin | = | REVIEW |
| BUG | PROJ-1 | add item 1 | Alana Grant | admin | = | DONE |



Another type of search is called basic search. In a basic search, issues are searched using a row of user-friendly interface fields, as shown here. Changes to these fields will change the resulting list of issues.

The screenshot shows the Jira Issues page for the 'projectA' project. At the top, there's a navigation bar with 'Share', 'Export issues', 'Go to advanced search', 'LIST VIEW', 'DETAIL VIEW', and a three-dot menu. Below the navigation is a search bar with a magnifying glass icon and a 'Save filter' button. A blue callout bubble with an 'i' icon points to the 'JQL' tab in the search bar, which is highlighted in blue. The main area displays a table of issues:

| Type | Key | Summary | Assignee | Reporter | P | Status |
|------|---------|-------------------|------------|---------------------|---|---------|
| PROJ | PROJ-13 | Submit blog post | Unassigned | Automation for J... | = | BACKLOG |
| PROJ | PROJ-12 | Approve blog post | Unassigned | Automation for J... | = | BACKLOG |
| PROJ | PROJ-11 | Create graphics | Unassigned | Automation for J... | = | BACKLOG |

Advanced search is an alternative to basic search, where you can specify your search using Jira Query Language, or JQL. This is a way to execute more complex searches than with basic search. We won't cover Advanced search in this course, but you can learn more about Advanced search and JQL in the Realizing the Power of Jira Reporting and Dashboards course.

Searching

Quick search

Basic search

Advanced search

Filters

Quick filters

Saved filters

Save a new filter

| Type | Key | Summary | Assignee | Reporter |
|-------|--|------------|----------|----------|
| APP-6 | Payment method (more info needed) | Unassigned | Ryan Lee | |
| APP-5 | As a user, I can add a comment | Unassigned | Ryan Lee | |
| APP-4 | As a user, I can set a profile picture | Unassigned | Ryan Lee | |
| APP-3 | As a user, I can create an account | Unassigned | Ryan Lee | |
| APP-2 | As a user, I can logout | Unassigned | Ryan Lee | |
| APP-1 | As a user, I can login | Unassigned | Ryan Lee | |

Filters are used to limit the issues that are displayed in search results. These are the default filters provided by Jira. For example, if you click on the Open issues filter, the results will show all issues in all projects that have not been closed. Even though filters are usually presented as user interface elements, they have an underlying JQL statement. That JQL statement defines which issues to show.

Searching

Quick search

Basic search

Advanced search

Filters

Quick filters

The screenshot shows a Jira Kanban board titled "TK board". At the top, there are several navigation and filter options: a search bar, a "BK" button, an "Add people" button, and a "Quick filters" dropdown menu. Below these are two buttons: "Only My Issues" and "Recently Updated". The board itself has five columns: BACKLOG 1, SELECTED FOR DEV... 2, IN PROGRESS 1, DEVELOPMENT DONE, REVIEW 1, and DONE 1. Each column contains one or more issues, each with a small profile picture and a link. A blue callout bubble with the text "Quick filters" and an info icon points to the dropdown menu at the top right.



The final search topic we will discuss is quick filters. Quick filters are used to limit which issues are shown on a board. You can click on a user's avatar to see only issues assigned to that user. You can click Only My Issues to see the issues assigned to you. You can click Recently Updated to view only issues with recent changes. You can also add custom quick filters to the board. Like standard filters, quick filters have an underlying JQL statement.

Topics

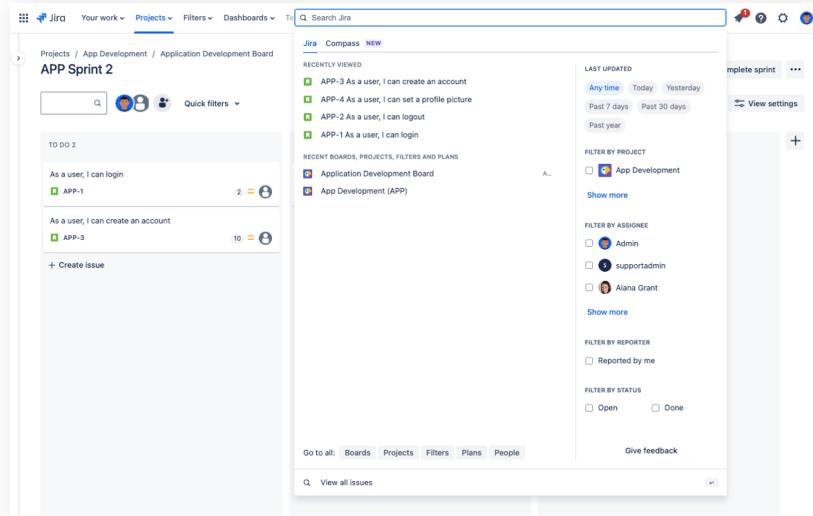
Searching overview

[Quick search](#)

[Basic search](#)



Quick search



When you initially open the quick search, the results show recent items, both issues and boards. You can click on an item to display it.

Quick search does not return just issues. When you open quick search, you can also access recent boards, projects, and filters. When you perform a search, all of these items that match will be displayed.

Quick search: search terms and keywords

The image shows three separate Jira search results, each with a numbered callout below it:

- 1**: A search for "feature". The results show six issues related to "feature":
 - add feature 2 PROJ-2
 - add feature 3 PROJ-3
 - add feature 1 PROJ-1
 - add feature 2 PRJ-2
 - add feature 3 PRJ-3
 - add feature 1 PRJ-1A "View all matching issues" button is at the bottom.
- 2**: A search for "feature NOT 1". The results show four issues related to "feature" that do not contain the word "1":
 - add feature 2 PROJ-2
 - add feature 3 PROJ-3
 - add feature 2 PRJ-2
 - add feature 3 PRJ-3A "View all matching issues" button is at the bottom.
- 3**: A search for "feature OR sample". The results show seven issues related to either "feature" or "sample":
 - add feature 3 PROJ-3
 - add feature 2 PROJ-2
 - add feature 3 PRJ-3
 - add feature 2 PRJ-2
 - add feature 3 PRJ-3
 - Instructions for deleting this sample SAM-17
 - add feature 1 PROJ-1A "View all matching issues" button is at the bottom.



Usually, you will just be entering simple text when searching with quick search. You also have the option of using logical operators within your search.

1. As an example, on the left, we are searching for the text "feature". You can see that this text is in the summary for six issues in our Jira account.
2. We can exclude feature one by adding a capitalized NOT followed by the number one. You can see that the two issue containing feature one are excluded from the results. In the search, NOT must be all caps. The search terms are not case sensitive, but text-field search keywords such as NOT, OR and AND must be capitalized.
3. In this example, we use the capital OR keyword to search for "feature" or "sample". For more information on this type of searching, see the link shown here on search syntax for text fields.

<https://support.atlassian.com/jira-core-cloud/docs/search-syntax-for-text-fields/>

Topics

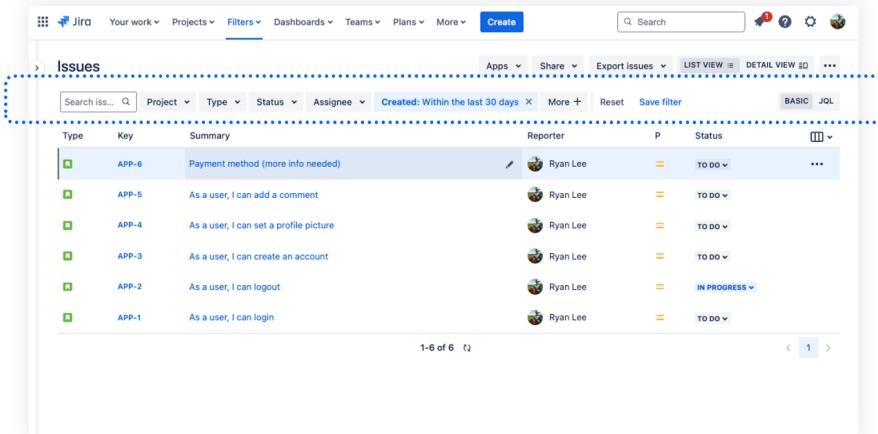
Searching overview

Quick search

Basic search



Basic search



The screenshot shows the Jira Issues navigator. At the top, there is a navigation bar with links like Jira, Your work, Projects, Filters, Dashboards, Teams, Plans, More, Create, and a search bar. Below the navigation bar is a toolbar with buttons for Apps, Share, Export issues, and view modes (List View, Detail View). A search bar is also present in the toolbar. A dashed blue rectangle highlights the toolbar area. The main content area is titled 'Issues' and contains a table with columns: Type, Key, Summary, Reporter, P, Status, and more. The table lists six issues, all reported by 'Ryan Lee'. The first issue is 'APP-6: Payment method (more info needed)'. The last issue is 'APP-1: As a user, I can login'. The status column shows various statuses: TO DO, TO DO, TO DO, TO DO, IN PROGRESS, and TO DO. The bottom of the table shows page navigation with '1-6 of 6' and a single page indicator.



Basic search uses user-friendly interface elements in the issue navigator to search for issues.

The row of elements in the top screenshot shows that we are in the basic search. You can see that the default search looks in all projects for issues of any type in any status and with any assignee, created in the last 30 days.

If you click on the project dropdown and select a project, you will see only the issues within that project.

Contains text

The screenshot shows the Jira Issues screen. At the top, there is a search bar with the text "user NOT login". A blue callout bubble with the number "1" points to this search bar, labeled "Search within issue text fields". Below the search bar, there are several filters: "Project", "Type", "Status", "Assignee", "Created: Within the last 30 days", "More +", "Reset", and "Save filter". The "Basic" tab is selected. The main area displays a table of issues with columns: Type, Key, Summary, Reporter, P, Status, and an ellipsis column. There are four issues listed:

| Type | Key | Summary | Reporter | P | Status | ... |
|------|-------|--|----------|---|-------------|-----|
| BUG | APP-5 | As a user, I can add a comment | Ryan Lee | = | TO DO | ... |
| BUG | APP-4 | As a user, I can set a profile picture | Ryan Lee | = | TO DO | ... |
| BUG | APP-3 | As a user, I can create an account | Ryan Lee | = | TO DO | ... |
| BUG | APP-2 | As a user, I can logout | Ryan Lee | = | IN PROGRESS | ... |

At the bottom of the table, it says "1-4 of 4".



Notice that basic search includes a text box. This behaves much like the quick search but is limited to searching issues. If you enter text in this textbox, Jira will search in fields that typically hold text, such as summary or description.

In this example, we entered "user NOT login" to the search field. This will filter the user stories with login keyword out of the results, and you can see that we are left with four issues. You can see that logical operators can be used in this field.

Searching more fields

The screenshot shows the Jira Issues search interface. At the top, there are several search filters: 'user NOT logout', 'Project', 'Type', 'Status', 'Assignee', and 'Created: Within the last 30 days'. Below these, a 'More +' button is highlighted with a blue circle containing an 'i' icon, which is connected by a vertical line to the text 'Add more fields to filter on'. The main search results table lists four issues: APP-5, APP-4, APP-3, and APP-1, each with a summary description. To the right of the table is a 'More' dropdown menu titled 'Search' with a magnifying glass icon. This menu contains a list of fields with checkboxes: 'Created' (checked), '[CHART] Date of First Response' (unchecked), '[CHART] Time in Status' (unchecked), 'Actual end' (unchecked), 'Actual start' (unchecked), 'Affects versions' (unchecked), and 'Approvals' (unchecked). At the bottom of the dropdown are 'Clear selection' and '11 of 71'.



If you want to search for values within fields not listed in basic search, you can click on the More dropdown to add searches for other fields.

Example - specifying a priority

The screenshot shows the Jira 'Issues' screen. At the top, there is a search bar containing 'user NOT logout' and several filter buttons: Project, Type, Status, Assignee, Priority, More +, Reset, and Save filter. A blue callout arrow points from the text 'Filter on the priority field' to the 'Priority' button. A dropdown menu is open over the 'Priority' button, titled 'Search'. It contains five options: 'Highest' (indicated by a red triangle pointing up), 'High' (indicated by a red triangle pointing up), 'Medium' (indicated by a yellow triangle pointing right), 'Low' (indicated by a blue triangle pointing down), and 'Lowest' (indicated by a blue triangle pointing down). The dropdown also includes a search bar and a '5 of 5' indicator at the bottom right.



Here, we have selected Priority from the More dropdown menu. Notice that Jira will automatically provide the appropriate interface depending on the type of field chosen. In this case, we have selected the Priority field, so Jira gives us a set of existing priority values.

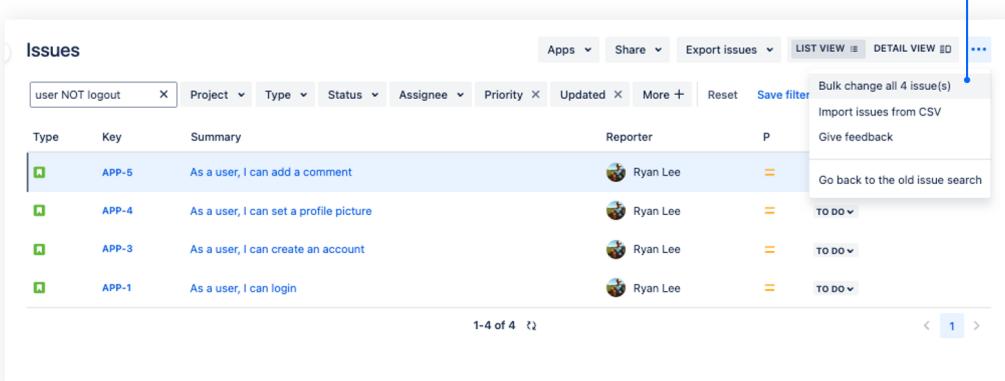
Example - specifying an Updated Date

The screenshot shows a Jira search results page titled 'Issues'. The search bar contains the query 'user NOT logout'. The 'Updated' filter is selected, opening a modal dialog. The dialog has four options: 'Within the last', 'More than', 'Between', and 'In the range'. The 'Between' option is selected, with date fields set to '4/14/2024' and '4/16/2024'. A blue 'Update' button is at the bottom of the dialog. The main search results table shows four issues: APP-5, APP-4, APP-3, and APP-1, each with a green icon and a brief summary.



Here, we have selected the Updated field, and you can see that Jira adds a dialog allowing you to specify the date values to search for. We have chosen to search for issues updated between two dates that we need to specify using the date calendars.

Bulk changing issues



The screenshot shows a Jira search results page for issues related to 'user NOT logout'. A context menu is open over the first issue, listing options: 'Bulk change all 4 issue(s)', 'Import issues from CSV', 'Give feedback', and 'Go back to the old issue search'. An annotation with a blue circle and an 'i' points to the 'Bulk change all 4 issue(s)' option.

| Type | Key | Summary | Reporter | P |
|------|-------|--|----------|---|
| BUG | APP-5 | As a user, I can add a comment | Ryan Lee | = |
| BUG | APP-4 | As a user, I can set a profile picture | Ryan Lee | = |
| BUG | APP-3 | As a user, I can create an account | Ryan Lee | = |
| BUG | APP-1 | As a user, I can login | Ryan Lee | = |



After performing an issue search, you can bulk change all issues within the issue search result, in this example, it's four issues.

Bulk changing issues is useful to correct a field value for a group of issues. Commonly, we find users make use of the bulk change feature to set the resolution of completed issues.

Are you getting it?



1. Which one of the following statements is true?
 - a. Quick filters are directly accessible from the global sidebar.
 - b. A filter and a quick filter are the same thing.
 - c. A filter has an associated JQL statement.



The answer is on the next slide.

Did you get it?



1. Which one of the following statements is true?
 - a. Quick filters are directly accessible from the global sidebar.
 - b. A filter and a quick filter are the same thing
 - ✓ c. A filter has an associated JQL statement..



Answer: c

Are you getting it?



2. How can you modify multiple issues at one time?
 - a. Execute a search for the issues that you want to change, then select "bulk change...".
 - b. Select multiple issues in the backlog, then edit a single issue.
 - c. Select multiple issues on a board, then edit a single issue.



The answer is on the next slide.

Did you get it?



2. How can you modify multiple issues at one time?
- ✓ a. Execute a search for the issues that you want to change, then select "bulk change...".
 - b. Select multiple issues in the backlog, then edit a single issue.
 - c. Select multiple issues on a board, then edit a single issue.



Answer: a

Takeaways



- Quick search can search the text of issues, board names, project names and filter names
- Basic search is a user-friendly way to search for issues



Lab 8 – Quick Search and Basic Search

- Perform quick searches
- Perform basic searches
- Work with search results
- Make bulk changes



9

Filters



What will you learn?

- Explore default filter queries
- Create a starred filter
- Explore quick filters
- Filter by label



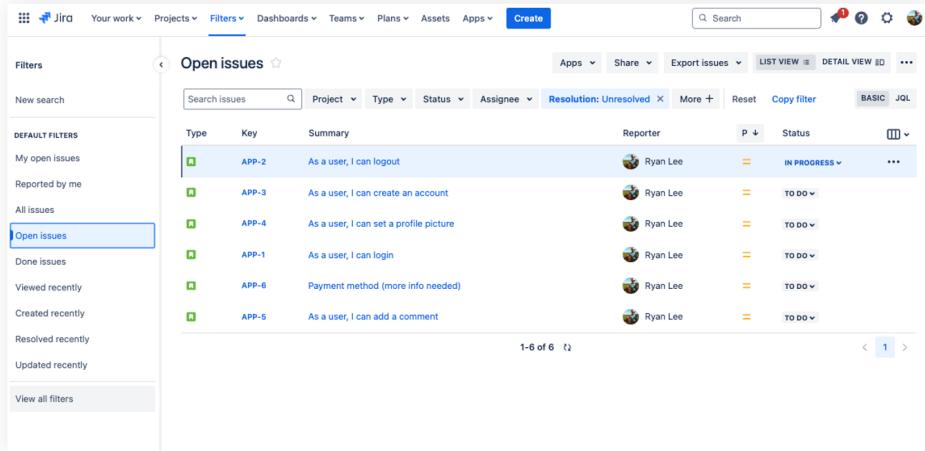
Topics

Filters

Quick filters



Filters



The screenshot shows the Jira interface with the 'Filters' sidebar open. The 'Open issues' tab is selected, highlighted with a blue border. The main area displays a list of six issues under the heading 'Open issues'. The columns are Type, Key, Summary, Reporter, P, Status, and more. The issues listed are APP-2 through APP-6, all reported by Ryan Lee and assigned to her.

| Type | Key | Summary | Reporter | P | Status |
|------|-------|--|----------|---|-------------|
| BUG | APP-2 | As a user, I can logout | Ryan Lee | = | IN PROGRESS |
| BUG | APP-3 | As a user, I can create an account | Ryan Lee | = | TO DO |
| BUG | APP-4 | As a user, I can set a profile picture | Ryan Lee | = | TO DO |
| BUG | APP-1 | As a user, I can login | Ryan Lee | = | TO DO |
| BUG | APP-6 | Payment method (more info needed) | Ryan Lee | = | TO DO |
| BUG | APP-5 | As a user, I can add a comment | Ryan Lee | = | TO DO |

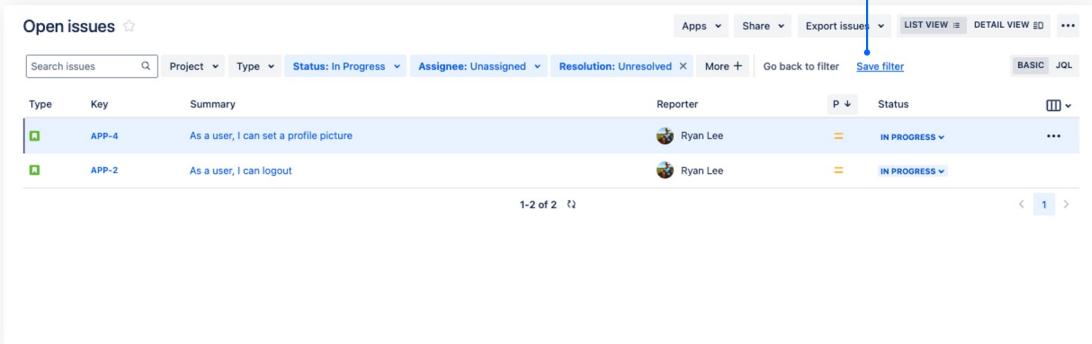


When using the issue navigator, you are presented with tabs in the sidebar that represent predefined searches, as shown in the image here.

Each of these predefined searches is a filter. Filters are a way to save and access searches that you commonly use.

Selecting a filter will execute the underlying JQL search and display the search configuration.

Save a search to create a filter



The screenshot shows a Jira search interface titled "Open issues". The search bar includes filters for "Status: In Progress" and "Assignee: Unassigned". Below the search bar, there's a table with columns: Type, Key, Summary, Reporter, P, Status, and three dots. Two issues are listed: APP-4 and APP-2, both reported by Ryan Lee and in the In Progress status.

| Type | Key | Summary | Reporter | P | Status | ... |
|------|-------|--|----------|---|-------------|-----|
| | APP-4 | As a user, I can set a profile picture | Ryan Lee | = | IN PROGRESS | ... |
| | APP-2 | As a user, I can logout | Ryan Lee | = | IN PROGRESS | ... |



Saving a search creates a filter. Start by creating and executing a query in either basic or advanced/JQL search, then click on the Save filter link to begin the process of saving the filter. In this example, we have created a search that displays all unassigned, unresolved issues in the In Progress status. We then select the "Save filter" link to create the filter.

Save the filter

The screenshot shows the Jira web interface. On the left, there's a sidebar with 'Filters' and 'DEFAULT FILTERS' sections. The main area is titled 'Issues' and shows a list of two items: 'APP-4' and 'APP-2'. A modal window titled 'Save filter' is open in the center. It has a required field 'Name*' with 'Unassigned In Progress' typed into it. Below it are optional fields for 'Description', 'Viewers' (set to 'Private'), 'Editors' (set to 'Private'), and a 'Save' or 'Cancel' button at the bottom.



We will name this filter "Unassigned In Progress".

The newly created filter

The screenshot shows the Jira interface for managing filters. On the left, there's a sidebar with various filter categories like 'My open issues' and 'Reported by me'. A blue dashed box highlights the 'Starred filters' section, which contains a single item: 'Unassigned In Progress'. The main panel displays the details for this filter:

- Filter details:** Shows the filter name 'Unassigned In Progress' with a star icon.
- Description:** 'This filter doesn't have a description.'
- Reporter:** Ryan Lee (Status: IN PROGRESS)
- Owner:** Admin (Status: IN PROGRESS)
- Permissions:** Visible to: Private, Editable by: Private.
- Subscriptions:** 'This filter doesn't have any subscriptions.'



Once you have created the filter, it will show in the sidebar under the "starred" category. You can click on the “Filter details” link to view and edit the details of the filter.

Edit filter details

The screenshot shows the Jira interface for managing filters. On the left, the 'Unassigned In Progress' filter is displayed with two issues listed: APP-4 and APP-2. The 'Filter details' tab is selected, showing sections for Description, Permissions, and Subscriptions. The 'Permissions' section indicates the filter is private. The 'Subscriptions' section shows no subscriptions. On the right, three overlapping dialog boxes are shown: 1) 'Update filter' dialog with fields for Name (Unassigned In Progress), Description, Viewers (private), Editors (private), and Save/Cancel buttons. 2) 'Filter Subscription' dialog with Recipients set to 'Personal Subscription', Schedule set to 'Daily', Interval set to 'once per day', and a note about the timezone. 3) A smaller 'Edit name and description' dialog is also visible.

Click on Filter details to view and edit the permissions and subscriptions related to the filter. By default, the filter is private, meaning it is only accessible to you. Subscriptions can be added to the filter, which emails recipients the query results at a rate specified in the subscription.

View all filters

The screenshot shows the Jira interface for managing filters. On the left, a sidebar titled 'Filters' contains sections for 'New search', 'DEFAULT FILTERS' (including 'My open issues', 'Reported by me', etc.), and 'STARRED FILTERS' (with 'Unassigned In Progress' highlighted). A dashed box surrounds the 'View all filters' link at the bottom of the sidebar. The main content area is titled 'Filters' and lists several filters with columns for 'Name', 'Owner', 'Viewers', 'Editors', and 'Starred by'. A 'Create filter' button is located in the top right corner.

| Name | Owner | Viewers | Editors | Starred by |
|-------------------------|----------|----------------------------|---------|------------|
| Filter for APPDEV board | Admin | App Development, All roles | Private | 0 people |
| Filter for PRJ board | Admin | projectB, All roles | Private | 0 people |
| Filter for PRJA board | Admin | projectA, All roles | Private | 0 people |
| Filter for PRJA1 | Ryan Lee | projectA, All roles | Private | 0 people |
| Unassigned In Progress | Admin | Private | Private | 1 person |

If you click on View all filters in the sidebar, you can see your newly created filter in the list. You can click on the more icon to the right of the filter to change metadata related to the filter.

Edit filter query

The screenshot shows the 'Filter details' screen in Jira. At the top, there are buttons for 'Revert to the original filter' and 'Save changes made to the existing filter'. Below this, the filter query is displayed: 'Unassigned In Progress'. The filter criteria are: Type: All standard issue types, Status: In Progress, Assignee: Unassigned, Resolution: Unresolved. The results table shows two issues: APP-4 and APP-2, both assigned to Ryan Lee and in IN PROGRESS status. A dropdown menu is open over the first issue, showing options: 'Save changes', 'Save as a copy', and 'As a user, I can logout'. The 'Save changes' option is highlighted.



To edit the query for a filter, execute the original filter so that you can view the query. Modify and execute the query. When the query is modified, you will see the Save filter dropdown and you can select Save changes to overwrite the existing query or Save as copy to create a new filter while leaving the original filter unchanged. You can also revert to the original filter query by clicking the Go back to filter link.

Topics

Filters

Quick filters



Filtering a board: quick filters

Further filters issues displayed on a board

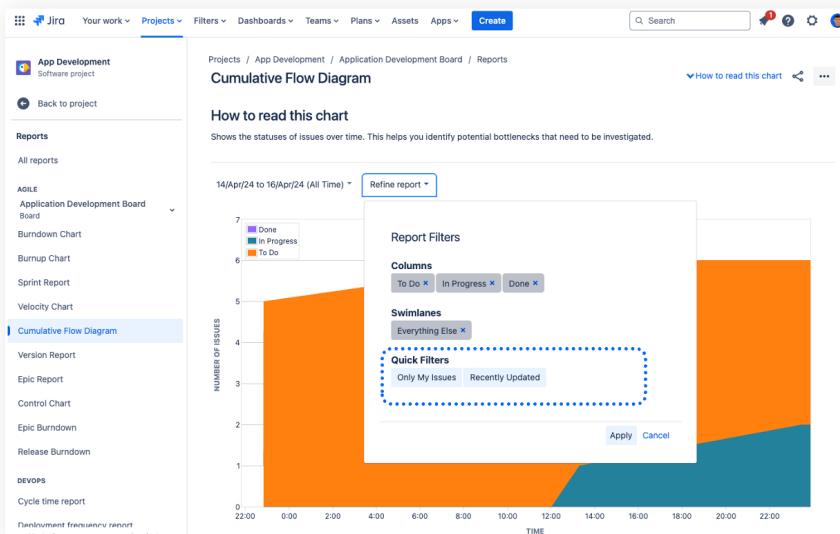
The screenshot shows the Jira interface for the 'App Development' project. On the left, there's a sidebar with navigation options like 'Planning', 'Active sprints', and 'Issues'. The main area is titled 'APP Sprint 2' and shows a Kanban board with three columns: 'TO DO 2', 'IN PROGRESS 1', and 'DONE'. Each column has two items, both of which are related to user authentication (login and logout). At the top of the board, there's a search bar and a 'Quick filters' dropdown menu. This menu is highlighted with a dashed blue box and contains two options: 'Only My Issues' and 'Recently Updated'. There are also icons for a user profile and a search function.



Quick filters are a way to further filter issues displayed on a board. By default, all the issues are displayed. You can see that there are two default quick filters. Check the Only My Issues box to limit the board to displaying issues in which you are the assignee. Check the recently updated checkbox to show only issues that have been updated recently. You can also use the search box to find issues on the board.

If you select a user icon, the board will only show issues assigned to that user.

Quick filters and reports



A board's quick filters can also be used to refine reports. Here we are viewing the cumulative flow diagram report. If we click on the "refine report" dropdown, we see that we can deselect columns or swimlanes to hide them from the report. We will discuss swimlanes a little later. We could further filter the issues that are being reported on by clicking on one or more quick filters to enable it. They are all not enabled by default. Be sure to click "Apply" to see the changes to the report.

Are you getting it?



1. Assume that a board shows 10 issues, and then you execute a quick filter. Which one of these statements is true?
 - a. There will be less than 10 issues on the board.
 - b. There will be more than 10 issues on the board.
 - c. There will be 10 or less issues on the board.



The answer is on the next slide.

Did you get it?



1. Assume that a board shows 10 issues, and then you execute a quick filter. Which one of these statements is true?
 - a. There will be less than 10 issues on the board.
 - b. There will be more than 10 issues on the board.
 - ✓ c. There will be 10 or less issues on the board.



Answer: c

Are you getting it?



2. Which one of the following statements is true?
 - a. Once a filter is created and saved, it can't be changed.
 - b. A filter is a saved search.
 - c. Filters you create can only be accessed by you.
 - d. You can edit and save changes to a filter.
 - e. You can edit and save a filter as a new filter.



The answer is on the next slide.

Did you get it?



2. Which one of the following statements is true?

- a. Once a filter is created and saved, it can't be changed.
- ✓ b. A filter is a saved search.
- c. Filters you create can only be accessed by you.
- ✓ d. You can edit and save changes to a filter.
- ✓ e. You can edit and save a filter as a new filter.



Answer: b, d, and e

Takeaways



- Filters are saved searches that can be exposed through user interface elements
- Quick filters are saved searches that are used to further limit the issues displayed on a board or in reports



Lab 9 – Filters

- Explore default filter queries
- Create a starred filter
- Explore quick filters
- Filter by label



10

Epics



What will you learn?

- Describe epics
- Work with epics
- Manage epics in the backlog



Topics

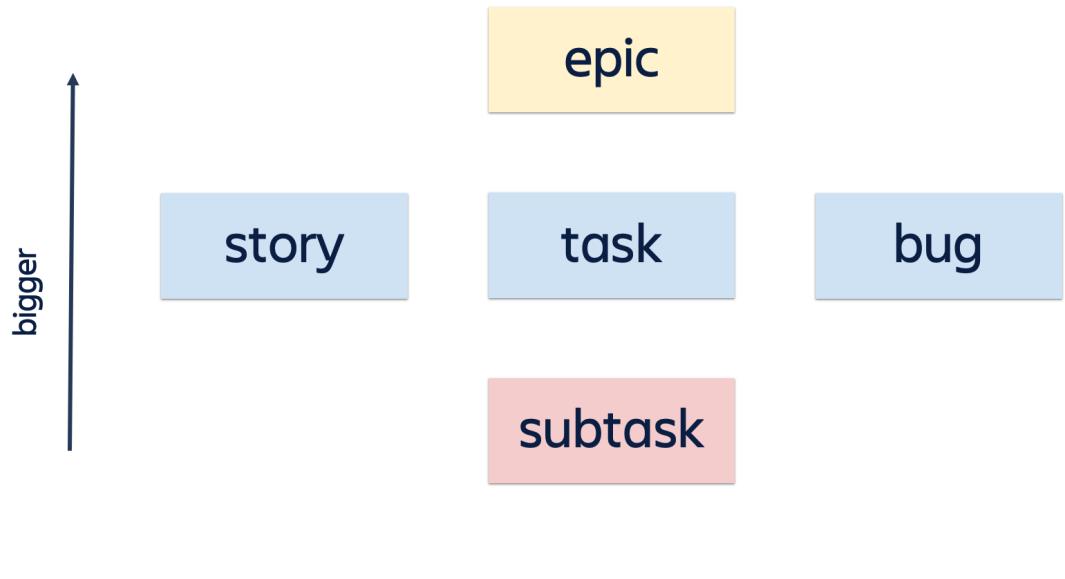
[Epics overview](#)

[Working with epics](#)

[Epics in the backlog](#)



Jira's issue type hierarchy



We have discussed stories, tasks, bugs and subtasks. An epic is an issue type that has a bigger scope than all of these. How they are actually used is up to the team.

Epic

- A large issue
- Can contain other issues
- Child issues can span multiple iterations, projects, teams and boards
- Can be a placeholder for many stories

Create issue

Import issues Configure fields ▾

Project*
projectA (PROJ)

Issue Type*
Epic

Some issue types are unavailable due to incompatible field configuration and/or workflow associations.

Epic Name*
Big Feature A

Provide a short name to identify this epic.

Summary*
add big feature A

Reporter*
Steve Byrnes

Start typing to get a list of possible matches.

Component/s
None

Create another Create Cancel



An epic is a large issue.

An epic can contain issues. They can contain stories, tasks, bugs, and custom issue types.

Child issues of an epic can span multiple iterations, projects, teams, and boards.

It can serve as a placeholder for many stories. For example, an epic might represent a feature in an app.

You then would break this feature into stories when the team can work on it.

Why epics?



Organization of work

Organize smaller work items and create relationships to higher level work items.



Multiple iterations

Span multiple iterations and projects.



Simplify

Simplifies backlog (one issue).



Epics are helpful because they organize the team's work rather than grouping unrelated stories.

They are also helpful if the work spans multiple iterations.

Epics can help simplify the backlog by representing a lot of work can simplify the backlog. You don't want to create detailed plans for an epic too early.

Topics

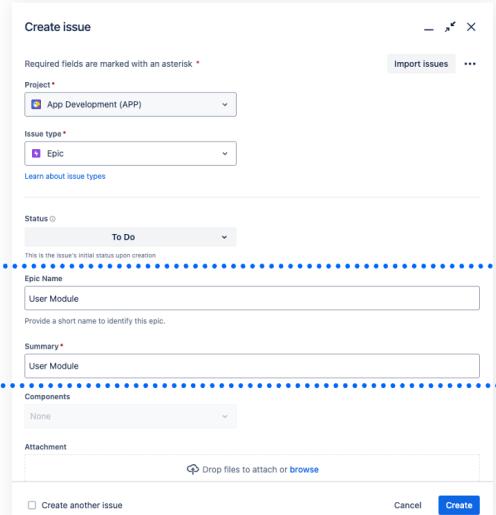
Epics overview

[Working with epics](#)

Epics in the backlog



Creating an epic



The screenshot shows the 'Create issue' dialog in Jira. At the top, it says 'Create issue' and 'Required fields are marked with an asterisk *'. There are dropdown menus for 'Project' (set to 'App Development (APP)') and 'Issue type' (set to 'Epic'). Below these are sections for 'Status' (set to 'To Do'), 'Epic Name' (containing 'User Module'), and 'Summary' (containing 'User Module'). A blue dotted box highlights the 'Epic Name' and 'Summary' fields. At the bottom, there are buttons for 'Create another issue' (unchecked), 'Cancel', and a prominent blue 'Create' button.



You can create an epic by selecting the Epic issue type when creating an issue. Here, we have created an epic named User Module.

Managing the issues of an epic

The screenshot shows a Jira interface for managing issues under an epic. The top navigation bar includes 'Projects / App Development / APP-7'. Below it, the 'User account module' header is visible. A blue dotted box highlights the 'Child issues' section, which lists one issue: 'APP-2 As a user, I can logout' (status: IN PROGRESS). The 'Activity' section below shows comments, with a placeholder 'Add a comment...' and a note about using 'M' to comment.



To add or view the issues of an epic, you can open the epic issue. Issues that belong to this epic are shown under “Child issues”.

Parent field

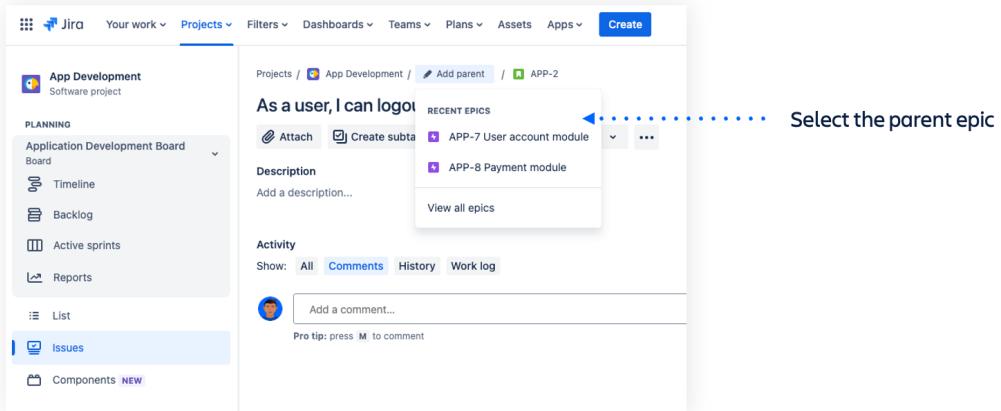
A field in the child issue pointing to the parent epic

The screenshot shows a Jira issue details page for an epic. The top navigation bar includes 'Projects / App Development / APP-7 / APP-2'. The main content area has tabs for 'Comments' (selected), 'History', and 'Work log'. A comment section with a placeholder 'Add a comment...' is visible. On the right, there's a 'Details' panel with fields like 'Assignee' (Unassigned), 'Reporter' (Ryan Lee), 'Sprint' (APP Sprint 2 +1), and 'Priority' (Medium). At the bottom of the details panel, under 'Epic Link', it says 'Replaced by Parent field'. A blue dashed box highlights the 'Parent NEW' link next to the epic name 'APP-7 User account mo...'. A small blue arrow icon is located on the right side of the page.

If you view an issue belonging to an epic, you will see the associated epic in the Parent field.
You can click on the link to view the epic.

Adding an existing issue to an epic

Open the child issue and select the existing epic under Epic Link

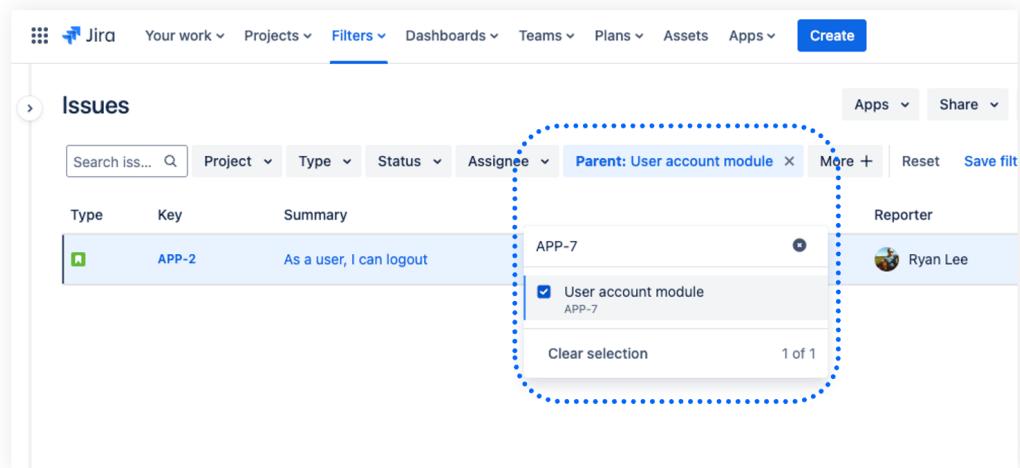


i The Epic link field has been replaced with the Parent field.



You also can add an existing issue to an epic at any time using the Parent field.

Searching for issues of an epic



The screenshot shows the Jira Issues search interface. At the top, there are navigation links: Jira, Your work, Projects, Filters (which is selected), Dashboards, Teams, Plans, Assets, Apps, and Create. Below the header is a search bar with placeholder text "Search iss..." and a magnifying glass icon. To the right of the search bar are buttons for Project, Type, Status, Assignee, and a dropdown labeled "Parent: User account module" which has a "X" icon next to it. Further right are buttons for More, Reset, and Save filt. On the far right of the header are Apps and Share dropdowns. The main area is titled "Issues" and contains a table with columns: Type, Key, and Summary. One row is visible: APP-7, APP-2, As a user, I can logout. A blue dotted circle highlights the "Parent: User account module" dropdown and its associated dropdown menu. This menu lists "User account module APP-7" with a checked checkbox, and below it are "Clear selection" and "1 of 1". To the right of the menu, there is a "Reporter" section showing a profile picture of Ryan Lee.



An epic is an issue type in Jira and, like all issues, has an issue key. If an issue is part of an epic, its Parent field contains the issue key of the epic issue. Here, we are searching for all issues in this epic. We can use Jira's autocomplete to help create this query. If you know the epic's key, you can enter that as shown above. If you know part of the Epic's name, you can also enter that.

Epic-based swimlanes

Issues under User account module epic

Issues without any epic assigned

The screenshot shows a Jira board titled "APP Sprint 2". At the top, there are navigation links for "Projects / App Development / Application Development Board". Below the title, it says "18 days remaining". The board has three main sections: "TO DO 2", "IN PROGRESS 1", and "DONE".

The "TO DO 2" section contains one issue: "As a user, I can logout" (User account module, APP-2).

The "IN PROGRESS 1" section contains one issue: "As a user, I can logout" (User account module, APP-2).

The "DONE" section is empty.

On the left side of the board, there are two categories:

- "Issues under User account module epic" (with a blue dotted arrow pointing to the first swimlane)
- "Issues without any epic assigned" (with a blue dotted arrow pointing to the second swimlane)

Below these categories, there are two more sections:

- "Issues under User account module epic" (with a blue dotted arrow pointing to the first swimlane)
- "Issues without any epic assigned" (with a blue dotted arrow pointing to the second swimlane)

Each section contains two issues:

- "As a user, I can login" (APP-1)
- "As a user, I can create an account" (APP-3)

At the bottom of each section, there is a "+ Create issue" button.

Here, we see an epic-based swimlane. Swimlanes are horizontal sections of boards containing issues with a common characteristic. Here, the issues of the epic are separated from the rest of the issues on the board.

NOTE: Your Jira or Board Administrator must configure the board to allow swimlanes.

Topics

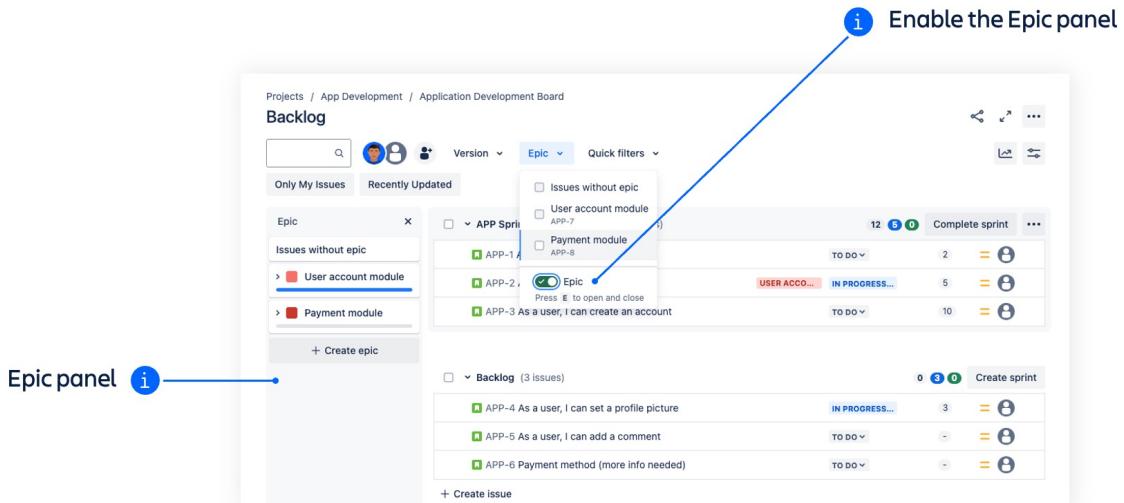
[Epics overview](#)

[Working with epics](#)

[**Epics in the backlog**](#)



Creating an epic from a backlog



When you toggle the Epic button, you choose to show the epics panel for the backlog. You can see all of the epics associated with the board.

You can select "Create epic" from the epics panel of the backlog to easily create a new epic.

Epics panel

The screenshot shows the Jira software interface with the following details:

- Left Sidebar (App Development Project):**
 - Planning: Application Development Board, Timeline
 - Backlog (selected)
 - Active sprints
 - Reports
 - List
 - Issues
 - Components NEW
 - Development: Code, Releases
 - Project pages
 - Add shortcut
- Top Bar:** Your work, Projects, Filters, Dashboards, Teams, More, Create, Search, Help, Settings, User icon.
- Central Panel (Backlog):**
 - Projects / App Development / Application Development Board
 - Backlog
 - Version, Epic, Quick filters
 - Only My Issues, Recently Updated
 - Epic:** APP Sprint 2 (15 Apr – 12 May) (3 issues)
 - APP-1 As a user, I can login (TO DO, 2)
 - APP-2 As a user, I can logout (IN PROGRESS..., 5)
 - APP-3 As a user, I can create ... (TO DO, 10)
 - Backlog:** (3 issues)
 - APP-4 As a user, I can set a p... (IN PROGRESS..., 3)
 - APP-5 As a user, I can add a c... (TO DO, 1)
 - APP-6 Payment method (mor... (TO DO, 1)
 - + Create issue



In the previous slide, you saw how to turn on the Epic panel. Once you have it turned on, there is a lot of information presented to you.

You can use the epics panel in the backlog to monitor the current status of specific epics. For example, you can quickly see the total number of issues in the epic and how many of those issues were actually completed.

Mark an epic as done

The screenshot shows the Jira Backlog interface for the 'App Development' project. On the left, the 'Epic' sidebar lists 'User account module' and 'Payment module'. A callout 'View epic details' points to the 'User account module'. In the main backlog area, an epic titled 'APP-2 As a user, I can logout' is shown with a status of 'IN PROGRESS...'. To its right, a detailed view of the 'User account module' epic is open, showing its workflow states: 'To Do', 'IN PROGRESS', and 'DONE'. The 'DONE' state is highlighted with a blue dashed border. The overall interface includes various filters, search bars, and navigation buttons.

Marking an epic's status as Done removes the epic from the Epic panel.

Are you getting it?



1. Which one of the following statements about epics is true?
 - a. They must contain other issues.
 - b. They can not contain other issues.
 - c. They can serve as a placeholder for many issues.



The answer is on the next slide.

Did you get it?



1. Which one of the following statements about epics is true?
 - a. They must contain other issues.
 - b. They can not contain other issues.
 - ✓ c. They can serve as a placeholder for many issues.



Answer: c

Are you getting it?



2. Which one of these is a common characteristic of issues in an epic?
 - a. Their "epic link" field has the same value.
 - b. Their "fixVersion" field has the same value.
 - c. Their "issue type" field has the same value.



The answer is on the next slide.

Did you get it?



2. Which one of these is a common characteristic of issues in an epic?

- ✓ a. Their "epic link" field has the same value.
- b. Their "fixVersion" field has the same value.
- c. Their "issue type" field has the same value.



Answer: a

Takeaways



- An epic is a large issue that may contain other issues
- The “Epic Link” field is used to associate other issues with an epic
- Epics can be shown on boards or in backlogs



Lab 10 - Epics

- Create an epic issue
- Add issues to an epic
- View an epic in the backlog
- Complete an epic



11

Dashboards



What will you learn?

- Describe dashboards
- Configure a dashboard



Visualizing work



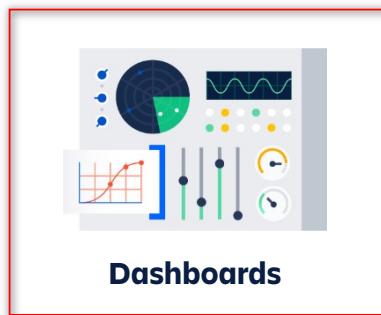
Boards



Search



Reports



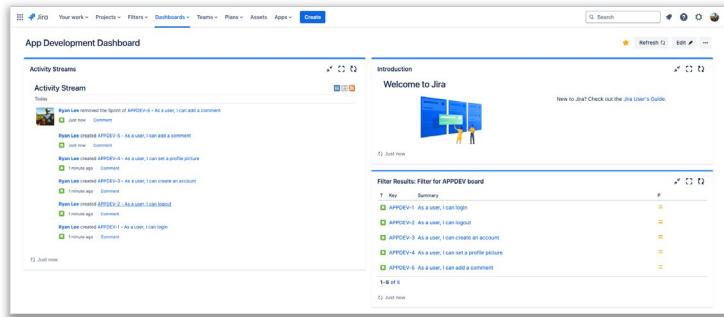
Dashboards



These are the main ways to visualize work using Jira. Here, we will discuss dashboards.

Dashboards

- Configurable view of the work of one or more projects
- Can be personal or shared
- Contains *gadgets*



Dashboards are a way to present a customized view of projects. They can be created for personal use or to share with others on the team. They are made up of one or more gadgets. A gadget displays aspects of the work on the projects.

Types of gadgets



Gadgets can be divided into categories:

Charts – Displays information visually using pie charts, bar charts, etc.

Jira – Present information about issues. The issues could be filtered by project, team member, status, etc. These can also be charts.

Wallboard – The information in these gadgets can be displayed in dashboards but is optimized for display as a wallboard.

Once placed on a dashboard, gadgets can be configured to customize your view of the Jira data.

ATLASSIAN Marketplace

Discover apps and integrations for Jira

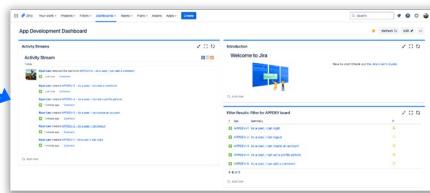
The screenshot shows the Atlassian Marketplace search results for 'Dashboard gadgets'. A red circle highlights the 'Dashboard gadgets' category in the dropdown menu at the top right of the search interface. Below the search bar, there are filters for 'Sort', 'Free for all teams', and 'More Filters'. The search results show 201 results for 'Dashboard gadgets'. The results are displayed in a grid format with three columns. Each result includes the app icon, name, description, rating, number of installs, and developer information.

| App | Description | Rating | Installs | Developer |
|--|--|-----------|----------------|----------------------------|
| Zephyr Scale - Test Management for Jira | A scalable, performant test management solution inside Jira with advanced test planning, reporting, and reusability features | ★★★★★ 333 | 27.7k installs | CLOUD FORTIFIED |
| Timelines by Tempo - Jira Time Tracking | New! AI-Powered Jira Time Tracking App for Effortless Timelines: #1 Jira App to report project health, billable hours &... | ★★★★★ 595 | 30.7k installs | CLOUD FORTIFIED |
| Structure by Tempo - Jira Portfolio Management & PPM | Jira Portfolio Management PPM Visualize All Data & Manage All Projects in Real-Time Manage Resources & Time in Status | ★★★★★ 299 | 13.4k installs | CLOUD FORTIFIED |
| Xray Test Management for Jira | Native Test Management. Built for every member of your team to plan, test, track and release great software | ★★★★★ 394 | 28k installs | CLOUD SECURITY PARTICIPANT |
| Custom Charts for Jira Reports and Time Status | Easy creation of comprehensive charts and reports for your Jira Dashboards. Report on various metrics including Time in Status | ★★★★★ 76 | 7.6k installs | CLOUD FORTIFIED |
| Rich Filters for Jira Dashboards | Dynamic and interactive dashboards, gadgets & real-time reports for Jira | ★★★★★ 109 | 5.8k installs | CLOUD FORTIFIED |



You can go to the Atlassian Marketplace, select Dashboard gadgets from the Categories dropdown, and see more gadgets you can add to dashboards.

Sharing dashboards



Ryan's Dashboard



Dashboards can be configured and used privately. They also can be shared with all or part of the team. Ryan created a dashboard that is very helpful in viewing the project's data for the sprint. Effie happened to be looking over Ryan's shoulder, saw the dashboard, and commented that she'd like to have a dashboard like that. Ryan can easily share the dashboard with Effie, and she now has the same view of the data as Ryan.

Are you getting it?



1. Which of the following statements about dashboards is true?
 - a. Dashboards can be configured.
 - b. Dashboards can be used privately.
 - c. Dashboards can be shared with all or part of the team.
 - d. All of the above.



The answer is on the next slide.

Did you get it?



1. Which of the following statements about dashboards is true?
 - a. Dashboards can be configured.
 - b. Dashboards can be used privately.
 - c. Dashboards can be shared with all or part of the team.
 - ✓ d. All of the above.



Answer: d

Takeaways



- Dashboards display the work of projects
- Dashboards can be shared or used personally
- Gadgets display a portion of a dashboard



Lab 11 - Dashboards

- Explore the default dashboard
- Create a dashboard



12

Putting it all Together



Topics

Quick course review

Jira family



Combined lean and agile principles

1. Empower the team

- Select motivated individuals
- Teams should self-organize
- Collaborate to create shared understanding

2. Visualize work

3. Experiment using the scientific method

- Continuously learn and improve
- Embrace change
- Partner with the customer
- Continuously inspect and adapt

4. Plan, develop and deliver incrementally

- Prefer conversations for conveying information
- Continuously refactor to maintain agility
- Maintain a sustainable pace
- Completed work items are the primary measure of progress
- Obtain fast feedback

5. Improve the "flow" of value

- Limit work in progress
- Map the value stream
- Pull work
- Eliminate waste
- Reduce setup times
- Automate what should be automated
- Continuously strive for simplicity

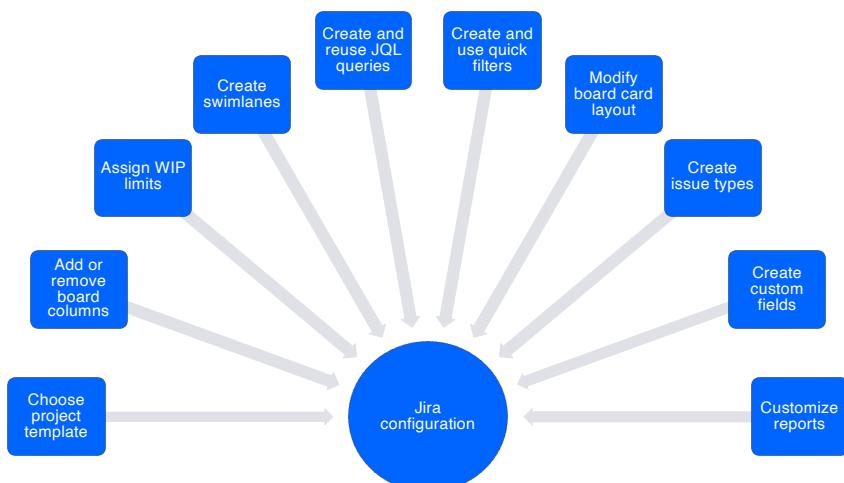
6. Build quality in

- Don't compromise on quality
- The process should identify problems
- Fix problems when they are discovered
- Identify and fix the root cause



Here are the combined lean and agile principles that we discussed. It's important that you and your team and your entire organization embody these principles to be effective. Having these principles, you can choose an agile methodology or framework as well as the tools to help you implement these principles.

Ways to configure Jira to match your team's process



We saw that an agile team is continuously changing and improving, and you want to understand Jira well enough to configure it to match your team's process. Jira is designed to be configurable for this very purpose.

Quick course review

Topics

Jira family



Jira family of products



Jira Work Management

Project management for marketing, HR, legal and other business teams



Jira Software

Plan, track and release software



Jira Service Management

Enables high-velocity request, incident and change management



Jira Product Discovery

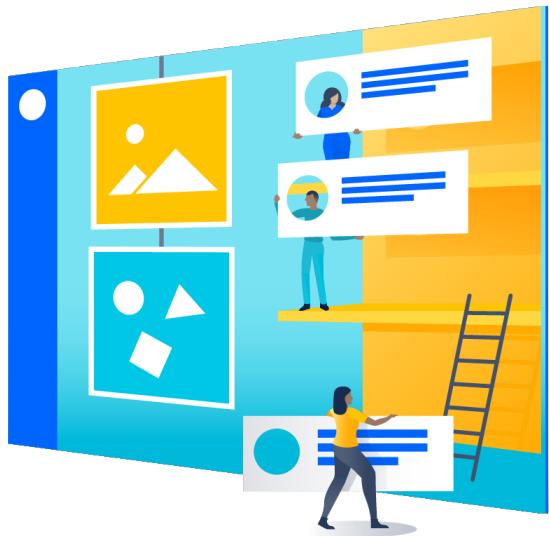
Capture and prioritize ideas to inform product roadmaps.



Jira Work Management, Jira Software, and Jira Service Management offer different project templates.

- Jira Work Management - Used to manage non-software projects and help keep teams organized. It allows you to manage projects, monitor details, and measure performance. Jira Work Management projects are included with Jira Software and Jira Service Management.
- Jira Software - Extends the functionality of Jira Work Management. Agile teams can plan, track, release, and report their progress while developing software. Includes scrum and kanban project templates. It also integrates with developer tools like version control systems and continuous delivery pipelines.
- Jira Service Management - Used by IT, ops, development, customer service, and other teams to handle requests, resolve incidents, and properly manage change. A company's work is visible across teams, enabling end-to-end service management and high-velocity delivery of services to customers.
- Jira Product Discovery - Used to capture and prioritize ideas in order to inform product roadmaps.

ATLASSIAN Marketplace



Can't find what you want for a gadget or a report out-of-the-box in Jira? Try the Atlassian Marketplace. There are over 1,000 apps available for just Jira. Someone may have already created what you need. Still can't find what you want? Jira has a rich UI where a developer can create a custom gadget.

Lab 12 – Capstone project (optional)

- Modify a Jira project to match your team's ever-changing agile processes



13

Lean and Agile Principles



What will you learn?



- Identify reasons the Toyota Production System is studied today
- Identify kanban objects
- Describe benefits of using kanban objects
- Identify kanban systems
- Describe lean principles
- Describe agile principles
- Compare lean and agile principles



Topics

Toyota Production System

Toyota kanban

Lean principles

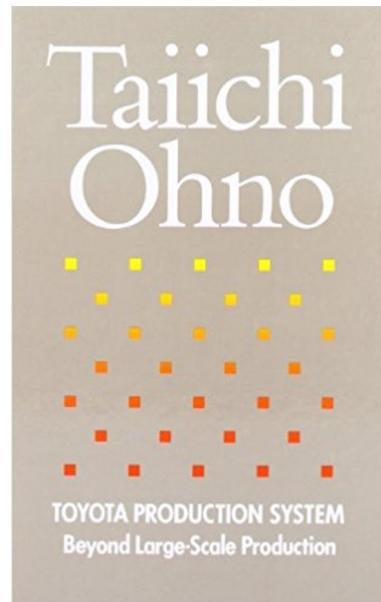
Agile Manifesto

Lean vs. agile



Toyota Production System

- Written in 1978, English translation in 1988
- Describes what is now sometimes called "lean thinking" or "lean management"
- Many agile principles are similar



Taiichi Ohno worked at Toyota and was one of the architects of the Toyota production system. In 1978, Taiichi Ohno wrote a book in Japanese, translated into English in 1988, about the thinking behind the system. In this book, Taiichi Ohno describes what is now sometimes called "lean thinking" or "lean management." We will see that this book contains many of the principles that are used in agile projects. Taiichi Ohno is considered to be the father of the Toyota production system,

Toyota simplified history

- "Catch up with America in three years."
- Focus was to eliminate waste and increase productivity
- Embraced ideas from Ford, but used a more "agile" approach

"I would like to emphasize that (the Toyota Production System) was realized because there were always clear purposes and needs."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



Here's a very simplified history of Toyota. After World War II, Japanese automakers were well behind US automakers, and they had a pretty ambitious goal to "catch up with America in three years" in the automobile industry. Especially after the war, Japan's automobile industry was very small compared to American companies like Ford and General Motors.

The main focus of the Toyota production system is to eliminate waste and increase productivity. Taiichi Ohno had heard that American workers were nine times more productive than Japanese workers and decided there must be a lot of wasted effort. The result was a decades-long focus on eliminating waste and continuously improving. This eventually resulted in Toyota becoming the largest automobile company in the world. Whether or not the productivity numbers were accurate, that's what Toyota believed and helps explain their focus on eliminating waste.

Toyota studied and embraced many of the production ideas from Ford but used a more "agile" approach to producing cars. After World War II, resources were very scarce in Japan, and Toyota couldn't afford to hold a lot of inventory as was done in traditional mass production. Out of need, Toyota developed a leaner, more flexible approach to mass-producing cars. We will see that this lean approach uses many of the same principles used today in agile projects.

In his book, Taiichi Ohno emphasized the need for the team to have a clear purpose, as the quote above demonstrates.

Lean principles apply broadly

"I am confident (the Toyota Production System) will reveal its strength as a management system..."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production
preface to the English edition (1988)



Lean principles of the Toyota production system can be applied in many contexts. They do not apply only to lean manufacturing or production. In fact, Taiichi Ohno wrote about this in the preface to the English edition of his book, which came out in 1988.

He said, "The Toyota production system is not just a production system. I am confident it will reveal its strength as a management system adapted to today's era of global markets and high-level computerized information systems."

Our discussion in this course focuses on lean principles that apply broadly.

Topics

Toyota Production System

Toyota kanban

Lean principles

Agile Manifesto

Lean vs. agile



What is kanban?

- Kanban - an object that controls the flow of work
- The idea came to Toyota from supermarkets
 - Instead of push, order when inventory is low (pull)
 - This matches the supply and demand
 - An empty box is a "kanban" - a signal to order more

"From the supermarket, we got the idea of viewing the earlier process in a production line as a kind of store."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



Kanban is a Japanese word used at Toyota to generically describe an object that controls workflow. The actual object can vary depending on the situation, but it usually holds some information related to the work. By naming this type of object a kanban, Toyota added a new definition to the word.

The idea came to Toyota from the way that supermarkets manage their inventory. Taiichi Ohno mentions in his book that he once came to the U.S. to visit car companies, but he was more excited about visiting and learning from the supermarkets.

In supermarkets, for some products, instead of forecasting and having a product delivered at regular intervals, which is a push system, more product is only ordered when the current inventory reaches a reorder point. This is a pull system.

This matches the supply and demand. For example, the supermarket doesn't need to guess how many cans of tomatoes to order and risk having too little or too much inventory. They can wait for one of the boxes to empty, then order another box.

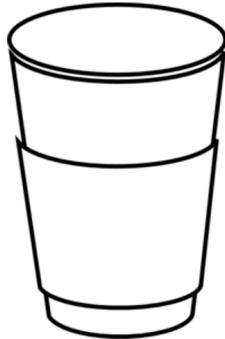
In this example, an empty box acts as a "kanban." It signals employees to order more product.

Taiichi Ohno said in his book, "From the supermarket, we got the idea of viewing the earlier process in a production line as a kind of store." When a later process used a component, such as an engine, it would provide a kanban to the earlier process. That kanban could be a piece of paper with information or something like an empty cart. This would signal the earlier process to build another component. The earlier process only builds a component when given a kanban. This is sometimes called a just-in-time system.

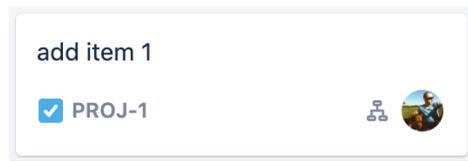
Other examples of kanbans



guest check



coffee cup



Jira issue



Even though the word kanban represents work to do at Toyota, the concept has been around for a very long time.

A guest check at a restaurant is a kanban. Preparing an order starts once the guest check is filled out. If no guests come in, no orders are prepared. The information related to the order is on the guest check. An empty coffee cup when ordering coffee can also act as a kanban. The size of the cup, your name, and the desired ingredients provide the information needed to start and fill your order. This kanban is also a convenient delivery vehicle for your order.

Another example of a kanban is a Jira issue. It provides information related to the work of the issue. The work shouldn't be started without the Jira issue in the correct column.

You can see that these kanbans represent and hold information related to the work to be done.

Kanban systems

"The Toyota Production System is the production method and the kanban system is the way it is managed."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



Projects / projectA / PROJ board

Kanban board

Only My Issues Recently Updated

| BACKLOG 1 | SELECTED FOR DEVELOPMENT 2 | IN PROGRESS 1 | REVIEW 2 | DONE 0 |
|----------------------|----------------------------|----------------------|----------------------|--------|
| add item 6 PROJ-6 | add item 4 PROJ-4 | add item 3 PROJ-3 | add item 1 PROJ-1 | |
| add item 5 PROJ-5 | | | add item 2 PROJ-2 | |

We're only showing recently modified issues.
Looking for an older issue?



Kanban is not used in isolation. There is an associated system to them, which sets the rules for how the kanban is used.

Taiichi Ohno said, "The Toyota production system is the production method, and the kanban system is how it is managed." He is pointing out that the kanbans are part of a kanban system.

In a restaurant, the kanban system is centered around guest checks.

The kanban system is centered around the work item on a project board.

Benefits of kanban systems

- Visualizes work
- Simple
- Reliable
- Efficient
- Eliminates waste
- Identifies bottlenecks/easy to improve

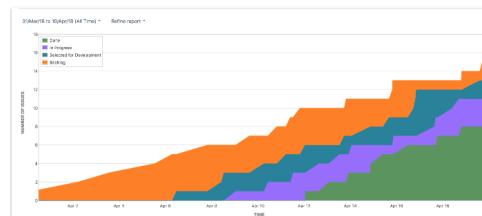


There are many benefits to using kanban systems to manage work. Some of the benefits include:

- They visualize the work. Kanbans are visual, allowing you to clearly see what work needs to be done.
- They are simple. Forecasting demand and maintaining inventories is usually a much more complicated approach to managing the work than using kanbans.
- They are reliable. This reliability comes from the visual aspect and simplicity of the system. A simple visual system is less likely to fail.
- They are efficient. An effective kanban system uses resources only to do work that is of value to the customer.
- They eliminate waste. Instead of holding and managing inventory, the supply is matched to the demand.
- Finally, it is easy to identify bottlenecks or problems in the system and make adjustments to improve the workflow. This is mainly because the simplicity and just-in-time nature of the system make it easy to see where problems occur. For example, many empty coffee cups indicate a problem.

Summary - kanban definitions

- Kanban token - An object that controls the flow of work
- Kanban system - A system that controls the flow of work using kanbans
- Kanban method - A lightweight agile method



The term kanban has a few meanings, depending on the context.

Kanban may refer to an object that controls the flow of work. We have seen that restaurant guest checks and Jira issues can be considered kanbans. We could call these kanban tokens to clarify the meaning.

Kanban may also refer to the system that controls the flow of work using kanbans. We have seen that Toyota has a system for using kanbans, just as restaurants have a system for using guest checks and how a team uses Jira issues. We could call this a kanban system to clarify the meaning.

We have also seen a lightweight, agile method called kanban. It takes its name from the kanban system at Toyota but applies it to agile projects. This needs to be clarified because kanban tokens, kanban systems, and even things like kanban boards are used in most agile methods, not only in the kanban method.

Are you getting it?



Which one of the following statements is true?

- a. Lean principles apply only to production.
- b. Lean principles are not related to agile principles.
- c. Many agile principles are similar to lean principles.



The answer is on the next slide.

Did you get it?



Which one of the following statements is true?

- a. Lean principles apply only to production.
- b. Lean principles are not related to agile principles.
- ✓ c. Many agile principles are similar to lean principles.



Answer: c

Topics

Toyota Production System

Toyota kanban

Lean principles

Agile Manifesto

Lean vs. agile



Why principles?

"With a better tool, we can get wonderful results. But if we use it incorrectly, the tool can make things worse."

"We should not forget to always use the principles..."

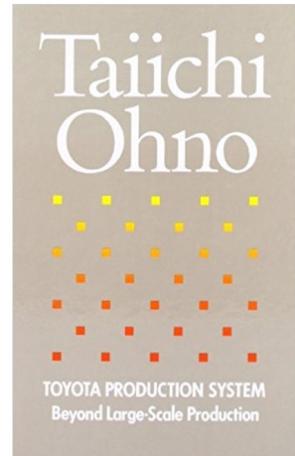
Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



We will look at lean principles because, in many ways, the principles are more important than the practices or tools. Principles are foundational and long-lived, even as the methods and tools change. Taiichi Ohno discusses the importance of understanding the principles in his Toyota Production System book, as the quotes above demonstrate.

Lean principles

1. Empower the team
2. Visualize work
3. Embrace the scientific method
4. Improve the "flow" of value
5. Build quality in



Here are some of the lean principles discussed in Taiichi Ohno's book. We will go through these one at a time, and you will see that we have already discussed many of these earlier in the course.

This is a subjective list. You might read his book and come up with a different list.

Toyota only invented a few of these principles, but their combination can and have been used effectively for many projects and businesses.

Empower the team

"...Operators acquire a **broad spectrum** of production skills... and **participate in** building up a total system in the production plant. In this way, the individual can **find value in working.**"

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



One of the fundamental principles of lean is to empower and leverage the creativity and talent of the team. This is not only smart business but also improves overall team happiness.

These two sentences contain some key concepts related to the people on the team.

First, they have diverse skills, including an understanding of lean principles. This differs from a traditional mass production line or a command-and-control-led team, where team members tend to have specialized jobs. Acquiring this diverse set of skills means the team members are always learning. Taiichi Ohno also mentions that team members participate in building the system. This means the team is trusted to make decisions and can even stop the production line if necessary.

Finally, empowering the team is good for the company's success and improves team satisfaction or morale. Many people want to feel like they are contributing to something of value, and empowering individuals provides that even more than monetary or recognition rewards.

Empower the team - teamwork

"A championship team combines good teamwork with individual skill."

"In modern industry, harmony among people in a group, as in teamwork, is in greater demand than the art of the individual craftsman."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



Teamwork is a big part of empowering the team.

Taiichi Ohno thought that we could learn a lot from team sports, and he mentions in his book that "A championship team combines good teamwork with individual skill."

Visualize work

- Visual control - Toyota uses kanbans to signal and control the work
- Andon board - An information board that shows any existing problems

"When one looks up, the andon (the line stop indication board) comes into view, showing the location and nature of trouble situations at a glance."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



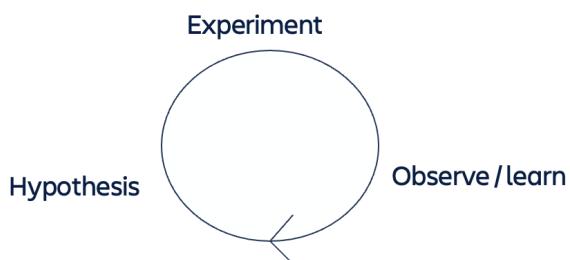
A principle that we have already discussed is to visualize work. We have seen that the work of agile projects is usually visualized with boards, and the team's progress is visualized with charts.

Toyota uses kanbans to signal and control the work. We have discussed kanbans and their benefits earlier.

Toyota also has something called an andon board. This is an information board that shows any existing problems.

Embrace the scientific method

1. Create a hypothesis
2. Build an experiment
3. Observe/learn from the results
4. Repeat/iterate



We have seen that managing and working on agile projects includes many small iterative learning loops. This resembles the scientific method, which can be considered the foundation of agile projects. Since most of us are at least familiar with the scientific method, we can use that knowledge to help understand agile projects.

The basic idea of the scientific method is to start with a theory or hypothesis, then build an experiment to test the theory, then observe the experiment's results and learn from it. You can then repeat the process, using what you have learned to improve during the next iteration. It is a series of continuously learning and improving iterations, which could also be called loops or cycles. This is called an empirical approach to problem-solving, where you are learning from the results of experiments.

In agile projects, we shift our thinking a little so that instead of using the scientific method only for discovery, we also use it to accomplish the work of a project. In addition to creating a hypothesis, we plan some work. We then do the work. We then learn from the feedback on our work and the process used to do the work. We repeat the cycle to do more work, leveraging the knowledge we gained in the previous iterations.

The scientific method is a formalized description of what the brain is doing when problem-solving. Accomplishing the work of projects involves a lot of problem-solving, so it is not surprising that agile approaches are based on the scientific method.

Embrace the scientific method

"The Toyota Production System has been built on the practice and evolution of this scientific approach."

"Progress can not be generated when we are satisfied with existing situations."

"... the new market demanded a constantly improving automobile."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



The scientific method, or similar experimental, evidence-based methods such as plan-do-check-adjust cycles, is used to continuously improve. We have already seen that the scientific method is the foundation of agile methods and is also true with lean thinking.

Every organization should be a learning organization. Every team should be a learning team. Everyone should realize that, even if they are successful now, they have much to learn.

A big focus at Toyota has been to improve continuously. Here is a quote from Taiichi Ohno's book that helps demonstrate that.

"...the new market demanded a constantly improving automobile." If you think about it, many markets, such as markets for cell phones and software, assume constantly improving products.

Embrace the scientific method - embrace change

"As long as we can not accurately predict the future,
our actions should change to suit changing situations.
In industry, it is important to enable people to cope
with change and think flexibly."

"Build a fine-tuning mechanism into the business so
that change will not be felt as change."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



A lean principle is to embrace change rather than create and stick to a fixed process. The second quote points out that change feels normal in a lean or agile environment and should be built into the company's process.

Improve the flow - limit work in progress / small batch size

"Reducing the number of kanban increases their sensitivity."

"People prefer working with large quantities. It is easier than having to work hard and learn from producing small quantities."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



Earlier, we discussed work-in-progress limits and described how limiting the amount of current work can increase the flow of value and overall productivity.

Here are some quotes from Taiichi Ohno related to limiting work in progress. One way to improve flow is to limit work in progress. He said reducing the number of kanbans increases their sensitivity. This means that less work-in-progress problems show themselves quite clearly. He also said people prefer working with large quantities. It's easier than working hard and learning from producing small amounts. This is a problem with mass-producing automobiles. With mass producing the features of a product, problems can be hidden in the volume of what you're doing and only be discovered much later. If you limit the work in progress, you get constant feedback, and you tend to see problems easier because they are noticed in a sea of other work.

Some people like the comfort of not getting consistent feedback because it is easier to continue doing what you are doing.

Improve the flow - map the value stream

1. Draw the current state
2. Draw the desired future state
3. Iterate toward the future state

https://en.wikipedia.org/wiki/Value_stream_mapping



Another way to improve flow is to map the value stream. Value stream mapping is a tool used for continuous improvement. Toyota calls this "material and information flow mapping."

You draw a diagram with the steps involved in the process that you want to improve. These often are the steps in bringing a product to the customer. This is called the current state.

You then identify the desired future state. This is done by finding ways to increase value and reduce waste. This often includes finding better ways to do things. The organization's vision and overall direction help to identify the desired future state.

You then iterate toward the future state. You may never fully reach the future state, but iterating towards it is a way to continuously improve. You can see that this is a learning loop that is similar to the scientific method. It is also similar to the agile approach of building the product incrementally. This simple exercise can be very good for planning and getting the team working in the same direction.

The Wikipedia link provides more information on value stream matching.

Improve the flow - pull work

"The conventional way was to supply materials from an earlier process to a later process. So, I tried thinking about the transfer of materials in the reverse direction."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



We have discussed pulling work from the previous step rather than having work pushed on you. This allows you to limit inventory and manage the flow of value. Taiichi Ohno discusses this in the quote above. He was a big proponent of using this kind of unconventional thinking to help solve problems.

Improve the flow - eliminate waste

"The basis of the Toyota Production System is the absolute elimination of waste."

"The vicious cycle of waste generating waste hides everywhere..."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



One of the most important principles of lean is to eliminate waste. Here are some quotes related to eliminating waste.

Taichi Ohno gives some examples of waste generating waste. For example, having too much inventory is a waste. It could cause the company to need a storage facility. This requires a system to manage the inventory. It also requires methods to ensure the inventory does not degrade before it is used. While this is a physical example, this concept of waste generating waste is true with other types of projects. For example, having too many features planned in detail is a waste because some features may never be built.

Improve the flow - reduce setup times

"Our production slogan is 'small lot sizes and quick setups'."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



One way to reduce waste is to reduce setup times. Toyota has flexible factories, allowing small batches of various products to be manufactured daily. For small batches to be practical, the tooling has to be fast.

That's why Taiichi Ohno said, "Our production slogan is 'small lot sizes and quick setups.'

Reducing setup times is essential in all types of work so you can focus on the actual work. Reducing setup times often adds flexibility that can change how the team works. If you work in the software field, moving to cloud computing reduces setup time, allowing you to quickly dynamically create computing instances, use them, and destroy them. This will enable you to quickly introduce code into production without batching features in two separate stages.

Improve the flow - automate what should be automated

"With computers available, it is a waste to perform calculations by hand."

Taiichi Ohno

Toyota Production System: Beyond Large-Scale Production



Another way to improve flow is to automate what should be automated. Taiichi Ohno said, "With computers available, it is a waste to perform calculations by hand." There are many examples where processes can be automated, and investing in that automation is essential.

Toyota makes sure to automate where it makes sense. The same applies to any knowledge work tasks that can and should be automated.

Summary - ways to improve the flow of value

1. Limit work in progress / small batch size
2. Map the value stream
3. Pull work
4. Eliminate waste
5. Reduce setup times
6. Automate what should be automated

"Look straight at the reality."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



Here is a summary of the ways to improve flow on any team.

Taichi Ohno said, "Look straight at the reality." Teams need to visualize their work and look into how the flow of value can be improved using some of the techniques shown here.

Build quality in

"...produce quality products 100 percent of the time..."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



A fundamental lean principle is to build quality into the product as you are building it. You don't want to start with poor quality and then have the mindset that the quality will improve over time.

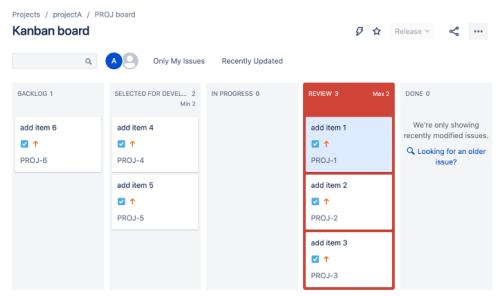
As Taiichi Ohno says, "produce quality products 100 percent of the time". It's essential to start with a solid technical foundation of what you are doing and to never compromise on quality to save time.

Thinking that being lean or agile means sacrificing quality is a misconception.

Build quality in - the process should identify problems

"...distinctions between normal and abnormal operations must be clear and countermeasures (solutions) always taken to prevent recurrence."

Taichi Ohno
Toyota Production System: Beyond Large-Scale Production



Related to building quality in is the principle that the process should identify problems. You want problems to be noticed.

This is one of the reasons for having the kanban board at Toyota. If there is a problem, it is displayed for all to see and take appropriate action.

We have also seen an example of identifying a problem by adding work-in-progress limits to columns on the board. When the team goes over the work-in-progress limit, the column is turned red to signal the team that there is a problem and to take appropriate action. Your team will have custom ways of identifying problems.

Build quality in - fix problems when they are discovered

"Correct a mistake immediately- to rush and not to take time to correct a problem causes work loss later."

Taichi Ohno
Toyota Production System: Beyond Large-Scale Production



When building quality in, fixing problems when they are discovered is crucial. You want to avoid building a list of problems to fix. This takes more time and discipline, but the result is much better quality.

He said, "Correct a mistake immediately - to rush and not to take time to correct a problem causes work loss later." The sooner you can fix the problem, the better, and this shows the problem with the traditional approach of QAing a product at the end of the life cycle.

Example - fix problems when they are discovered



“Because a device that could distinguish between normal and abnormal conditions was built into the machine, defective products were not produced.”

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production

<http://www.tcmit.org/english/exhibition/textile/fiber03.html>



In his book, Taiichi Ohno describes the origin of the idea to fix problems when they are discovered. Sakichi Toyoda was the founder of the Toyota Motor Company. Prior to that, he owned a textile company that weaved fabrics. He invented an auto-activated weaving machine. The machine would stop if any of the threads broke. An operator could then replace the thread. Before this invention, if a thread broke, it was only discovered after the defective products were made. In other words, the quality control was moved from after the product was built to during the building of the product. If you are familiar with the concept of test-driven software development, you will notice that this is the same principle. Any problems with the software are discovered and fixed as the software is being developed.

Build quality in - identify and fix the root cause

"By asking why five times and answering it each time, we can get to the real cause of the problem, which is often hidden behind more obvious solutions."

Taiichi Ohno
Toyota Production System: Beyond Large-Scale Production



When problem-solving, it is essential to identify and fix the root cause of the problem.

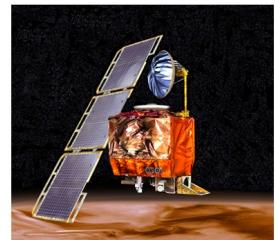
Taiichi Ohno talks about Toyota's "five whys" problem-solving approach.

The idea is to continue asking why questions until you find the root cause of the problem and then fix the problem there. The number five is a typical number of times the why question needs to be asked.

You want to keep asking why until you aren't talking about a symptom but the root cause of the problem. Stopping early and fixing a symptom usually means the root cause will create a problem somewhere else.

Example: asking "why" to find the root cause and solution

Problem: The orbiter crashed into the planet.



Question: Why did the orbiter crash into the planet?

Answer: Because it didn't have the proper trajectory on approach.

Question: Why didn't it have the proper trajectory on approach?

Answer: Because the thrusters did not work properly.

Question: Why didn't the thrusters work properly?

Answer: Because the acceleration data in the software was inaccurate.

Question: Why was the acceleration data inaccurate?

Answer: Because one team used metric units and the other used English units.

Solution: Standardize on a single system of measurement. Ensure pre-launch tests fail in similar circumstances.

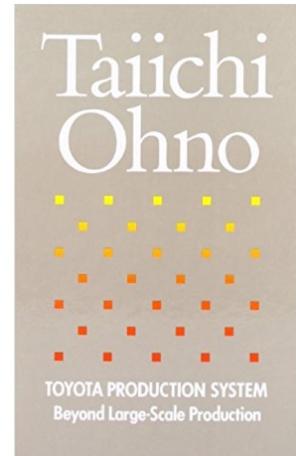


This example is based loosely on the 1998/1999 Mars Climate Orbiter.

https://en.wikipedia.org/wiki/Mars_Climate_Orbiter

Lean principles

1. Empower the team
2. Visualize work
3. Embrace the scientific method
 - a. Continuously learn and improve
 - b. Embrace change
4. Improve the "flow" of value
 - a. Limit work in progress / small batch size
 - b. Map the value stream
 - c. Pull work
 - d. Eliminate waste
 - e. Reduce setup times
 - f. Automate what should be automated
5. Build quality in
 - a. The process should identify problems
 - b. Fix problems when they are discovered
 - c. Identify and fix the root cause



That is an overview of some of the lean principles. Taken together, they provide a way of working that is very effective in many contexts.

If you look at the individual principles, you can see that they are timeless. The word "lean" is currently used to describe them, but that word is not important and may not be universally used. We will see that agile principles are very similar to lean principles.

Are you getting it?



An andon board is a central location that displays current problems that the team has identified. Which one of these is another example of the process of identifying problems?

- a. Moving an issue to a "queue" column on a board.
- b. Columns on boards changing color when over the work-in-progress limits.
- c. An issue that has been in the backlog a long time.



The answer is on the next slide.

Did you get it?



An andon board is a central location that displays current problems that the team has identified. Which one of these is another example of the process of identifying problems?

- a. Moving an issue to a "queue" column on a board.
- ✓ b. Columns on boards changing color when over the work-in-progress limits.
- c. An issue that has been in the backlog a long time.



Answer: b

Are you getting it?



When troubleshooting a problem using the "5 whys" approach, when should you stop asking "why" questions?

- a. When you find the root cause of the problem.
- b. When you have asked five questions.
- c. When you find the solution to your problem.



The answer is on the next slide.

Did you get it?



When troubleshooting a problem using the "5 whys" approach, when should you stop asking "why" questions?

- ✓ a. When you find the root cause of the problem.
- b. When you have asked five questions.
- c. When you find the solution to your problem.



Answer: a

Are you getting it?



Which one of these is an example of a team that uses an approach similar to the scientific method?

- a. The team creates a plan for building a product and then implements the plan perfectly.
- b. The team builds a feature, but the data shows that customers do not use it, so they remove it.
- c. The team postpones any changes to the plan until after the initial product release.



The answer is on the next slide.

Did you get it?



Which one of these is an example of a team that uses an approach similar to the scientific method?

- a. The team creates a plan for building a product and then implements the plan perfectly.
- ✓ b. The team builds a feature, but the data shows that customers do not use it, so they remove it.
- c. The team postpones any changes to the plan until after the initial product release.



Answer: b

Topics

Toyota Production System

Toyota kanban

Lean principles

Agile Manifesto

Lean vs. agile



Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

© 2001, the above authors
this declaration may be freely copied in any form,
but only in its entirety through this notice.

<https://agilemanifesto.org>



Here is the manifesto for agile software development, commonly called the Agile Manifesto. We will go through it piece by piece.

First of all, notice that the copyright statement is from 2001. The authors agreed at the time to keep it moving forward, so it is a static, historical document. Having a static record related to agile software development is ironic because continuous improvement is fundamental to agility. The document is static, not because the writers thought it was perfect, but because this is what they agreed to at that time. The agile manifesto represents the commonality of some of the different agile approaches at the time.

If we look back at the Agile Manifesto's title, we can see that it was intended as an approach to software development. Agility is broader than that, so the scope of this document is limited. However, we will see that many ideas apply to agile projects in general.

The agile manifesto starts with some context setting. "We are uncovering better ways of developing software by doing it and helping others do it."

The "we" in that sentence refers to the people who authored the Agile Manifesto. They had much experience and described what worked well in developing software.

By "uncovering better ways of developing software," they are saying that in their experience, the plan-based or waterfall approach does not work well for software development compared to an agile approach. When they wrote this, agile methods were not as common as they are today, and this was an attempt to help transform the thinking of the software development industry.

Next, the manifesto contains four value statements. The first one is "through this work, we have come to value individuals and interactions over processes and tools." Using "over" in the value statements helps clarify their discussion. This approach makes more sense when convincing an industry to change from waterfall to agile. They place a high value on individuals and their interactions. This is related to empowering the team and encouraging

collaboration. They are placing lower value on processes and tools. An empowered team can choose the processes and tools that work best for them.

After the four value statements, they further explain the structure of value statements. "That is, while there is value in the items on the right, we value the items on the left more." Again, this makes sense when trying to convince an industry to change. In an agile approach, the team is empowered to make decisions. The processes and tools that this team uses are secondary to that.

The following statement is "... we have come to value working software over comprehensive documentation". This places the value on continuously building things that work over heavy upfront planning. This is similar to the lean principle of focusing on value and eliminating waste. A sizeable up-front planning document will likely need to be corrected and/or misunderstood and, therefore, is a source of waste.

The following statement is "... we have come to value customer collaboration over contract negotiation". This is saying that the customer should be a team member, realizing that building the project is a continuous learning process on all sides. You want to avoid an arms-length relationship with your customer bound by an inflexible agreement, as is common with waterfall projects. When an upfront agreement is made, the project's true nature is only partially known, and the team and its customers should realize that and work together.

The last of the four statements is "...we have come to value responding to change over following a plan". This is acknowledging that, especially in complex one-off projects, you can not predict the future, and instead of heavy upfront planning, a better strategy is to embrace change.

So, that is a look at the value statements in the Agile Manifesto. They said that they had experienced agile processes working better for software development than traditional waterfall processes. These ideas seem straightforward and almost common sense now, but many projects that should have been managed this way were not, and in some cases, that is true for current-day projects.

You can see that most of these value statements have nothing to do with software and can apply to many projects. For example, by changing the words "working software" to "incremental planning and development," the ideas are more generally applicable.

Agile Manifesto value statements - takeaway principles

1. Empower the team
 - "Individuals and interactions over processes and tools"
2. Embrace change
 - "Responding to change over following a plan"
3. Partner with the customer
 - "Customer collaboration over contract negotiation"
4. Plan, develop and deliver incrementally
 - "Working software over comprehensive documentation"



We will convert the Agile Manifesto value statements into principles to simplify things. We will then build an organized list of principles as we continue.

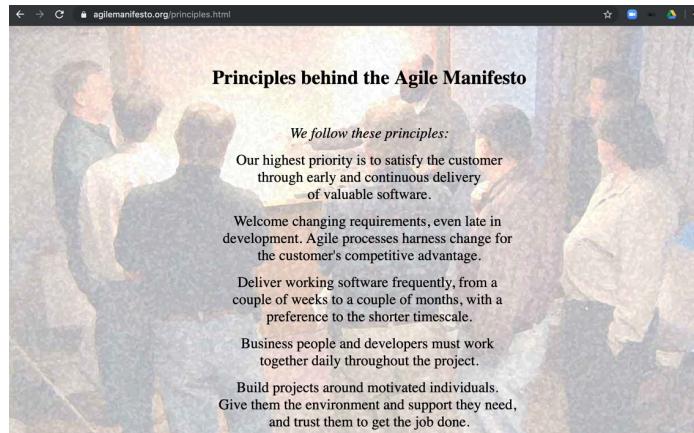
The first principle is to empower the team. This comes from the "individuals and interactions over processes and tools" statement.

The second principle is to embrace change. This comes from the "responding to change over following a plan" statement.

The third principle is to partner with the customer. This comes from the "customer collaboration over contract negotiation" statement.

The fourth principle is to plan and develop incrementally. This comes from the "working software over comprehensive documentation" statement.

Principles behind the Agile Manifesto



<https://agilemanifesto.org/principles.html>



Besides the four value statements, there are 12 principles behind the Agile Manifesto. The link above contains the principles.

Principles behind the Agile Manifesto

1. Empower the team
 - a. Select motivated individuals
 - b. Teams should self-organize
 - c. Collaborate to create shared understanding
2. Embrace change
 - a. Partner with the customer
 - b. Obtain fast feedback
 - c. Continuously inspect and adapt
3. Plan, develop and deliver incrementally
 - a. Prefer conversations for conveying information
 - b. Continuously refactor to maintain agility
 - c. Maintain a sustainable pace
 - d. Completed work items are the primary measure of progress
4. Focus on value
 - a. Eliminate waste
 - b. Continuously strive for simplicity
 - c. Don't compromise on quality



This is a combined list of principles from the Agile Manifesto value statement and 12 principles, rewritten for general project use.

Topics

Toyota Production System

Toyota kanban

Lean principles

Agile Manifesto

Lean vs. agile



Lean vs. agile

- Lean
 - Used at MIT by John Krafcik (1988)
 - Described the ideas of the Toyota Production System
 - Applies to any type of project
- Agile
 - Used by the participants who created the Agile Manifesto (2001)
 - Described a lightweight alternative to waterfall software development
 - Applies to any type of project
 - The terms are often used interchangeably
 - Can think of lean as more “foundational” than agile – you can be lean and use waterfall project management



We have discussed lean and agile principles.

Lean is a term used at MIT by John Krafcik.

It was used to describe the ideas of the Toyota Production System.

The ideas apply to any project, even personal projects.

Agile is a term used by participants who created the Agile Manifesto in 2001.

Agile originally described a lightweight alternative to plan-driven, or waterfall, software development.

Over time, agile has taken on a broader meaning that can be applied to any project.

The Toyota Production System could have been named agile; it's just that lean was chosen. Similarly, agile could have been called lean.

Lean and agile are often used interchangeably to describe an adaptive approach focusing on value.

Many common principles are found in Taiichi Ohno's 1978 book, "Toyota Production System."

Lean and agile principles

Lean

1. Empower the team
2. Visualize work
3. Embrace the scientific method
 - a. Continuously learn and improve
 - b. Embrace change
4. Improve the "flow" of value
 - a. Limit work in progress / small batch size
 - b. Map the value stream
 - c. Pull work
 - d. Eliminate waste
 - e. Reduce setup times
 - f. Automate what should be automated
5. Build quality in
 - a. The process should identify problems
 - b. Fix problems when they are discovered
 - c. Identify and fix the root cause

Agile

1. Empower the team
 - a. Select motivated individuals
 - b. Teams should self-organize
 - c. Collaborate to create shared understanding
2. Embrace change
 - a. Partner with the customer
 - b. Obtain fast feedback
 - c. Continuously inspect and adapt
3. Plan, develop and deliver incrementally
 - a. Prefer conversations for conveying information
 - b. Continuously refactor to maintain agility
 - c. Maintain a sustainable pace
 - d. Completed work items are the primary measure of progress
4. Focus on value
 - a. Eliminate waste
 - b. Continuously strive for simplicity
 - c. Don't compromise on quality



Here are the lean and agile principles that we have discussed. There is a lot of overlap, but also some differences. It's probably not necessary to differentiate lean and agile principles, so next, we will simplify this by combining the lists.

Combined lean and agile principles

1. Empower the team
 - a. Select motivated individuals
 - b. Teams should self-organize
 - c. Collaborate to create shared understanding
2. Visualize work
3. Experiment using the scientific method
 - a. Continuously learn and improve
 - b. Embrace change
 - c. Partner with the customer
 - d. Continuously inspect and adapt
4. Plan, develop, and deliver incrementally
 - a. Prefer conversations for conveying information
 - b. Continuously refactor to maintain agility
 - c. Maintain a sustainable pace
 - d. Completed work items are the primary measure of progress
 - e. Obtain fast feedback
5. Improve the "flow" of value
 - a. Limit work in progress / small batch size
 - b. Map the value stream
 - c. Pull work
 - d. Eliminate waste
 - e. Reduce setup times
 - f. Automate what should be automated
 - g. Continuously strive for simplicity
6. Build quality in
 - a. Don't compromise on quality
 - b. The process should identify problems
 - c. Fix problems when they are discovered
 - d. Identify and fix the root cause



Here is the combined list of lean and agile principles. These are all sound principles to understand well and to use where appropriate.

Are you getting it?



Which one of these Agile Manifesto value statements is most related to empowering the team?

- a. Working software over comprehensive documentation.
- b. Individuals and interactions over processes and tools.
- c. Customer collaboration over contract negotiation.



The answer is on the next slide.

Did you get it?



Which one of these Agile Manifesto value statements is most related to empowering the team?

- a. Working software over comprehensive documentation.
- ✓ b. Individuals and interactions over processes and tools.
- c. Customer collaboration over contract negotiation.



Answer: b

Are you getting it?



Which one of these statements is similar to the spirit of the Agile Manifesto?

- a. The product should only be delivered when it meets the customer's specifications.
- b. All communication should be documented.
- c. It is important to embrace change.



The answer is on the next slide.

Did you get it?



Which one of these statements is similar to the spirit of the Agile Manifesto?

- a. The product should only be delivered when it meets the customer's specifications.
- b. All communication should be documented.
- c. It is important to embrace change.



Answer: c