

Jira Software Questions and Answers



Table of content

1. Introduction to Jira Software
2. Jira Login and User Navigation
3. Jira Tickets: Creating and Managing Issues
4. Jira Bug Tracking and Issue Tracking
5. Jira for Agile Project Management
6. Jira Sprint Management
7. Jira Reporting and Metrics for Testers
8. Jira Automation Integration with Testing Tools
9. Customizing Jira Workflows
10. Defect Lifecycle Management in Jira

1. Introduction to Jira Software

Question: What is Jira? How is it used in software development projects?

Answer: Jira is a project management and issue-tracking tool developed by Atlassian. Originally featuring bug and issue tracking, Jira has evolved into a powerful tool used in Agile project management. It helps teams plan, track, release, and monitor software throughout the software development lifecycle.

For SDETs (Software Development Engineers in Test) and QA engineers, Jira provides the functionality for managing test cases, tracking defects, monitoring progress, and cooperating with the development teams. It works with Agile methodologies such as Scrum and Kanban, allowing teams to manage sprints, backlogs, and releases. For example, an SDET might use Jira to track the progress of automated tests, create bug reports based on test results, or link test cases to specific user stories or epics in a sprint.

Question: Why is Jira popular in the software development industry?

Answer: Jira is popular because it adapts to various workflows and works well for teams of all sizes. It supports Agile practices, allowing teams to manage backlogs, sprints, and releases easily. Jira also integrates with many tools used by software development teams, such as Confluence, Bitbucket, and testing frameworks, making it a central hub for tracking progress.

For QA teams, Jira simplifies bug tracking, test management (with plugins), and issue tracking. It's ability to generate reports and dashboards gives insights into the overall quality of the software.

Question: What are the Jira features that SDETs and QA engineers should know about?

Answer: Some of the key features that are useful for testing include:

- **Issue Tracking:** Jira allows users to create and track issues such as bugs, user stories, and tasks. This makes it easier to monitor the progress of testing, log defects, and link test results to development work. Example: A QA engineer can log a bug in Jira with detailed reproduction steps and assign it to a developer for fixing.
- **Custom Workflows:** Jira workflows help teams track the status of issues from start to finish. Each project can have a customized workflow that aligns with the team's processes. Example: A testing team may create a workflow where bugs move through statuses like "Open," "In Progress," "Fixed," and "Closed."
- **Test Case Management:** Although Jira doesn't have built-in test case management, it integrates with plugins like Zephyr and Xray. These tools help SDETs manage test cases within Jira, link them to user stories, and track execution results.
- **Dashboards and Reports:** Jira's dashboards and reports allow teams to visually view the project progress, defect statuses, and test case execution. QA engineers can create custom reports to monitor open bugs or track testing coverage. Example: A QA engineer can create a report that shows the number of open bugs in the current sprint.
- **Agile Boards:** Jira supports both Scrum and Kanban boards. These boards help teams organize their work, track progress, to know if all tasks are being executed as expected. Example: A Scrum team may use Jira's Scrum board to track user stories, bug fixes, and testing tasks in the sprint backlog.
- **Automation and Integration:** Jira integrates with Continuous Integration (CI) and Continuous Deployment (CD) tools like Jenkins. This allows automated test results to be linked directly to Jira issues, automatically updating their status based on test results. Example: When an automated test fails, it can trigger a Jira ticket or update an existing bug's status.

Question: How does Jira help QA engineers manage bugs and defects?

Answer: Jira's bug tracking system is easy to use and shows the status of the bugs. QA engineers can log bugs with detailed descriptions, screenshots, and logs. Each bug can be linked to a user story, assigned a priority, and tracked through the development process.

Jira also supports a complete defect lifecycle, with stages like "Open," "In Progress," "Resolved," and "Closed." This allows the team to monitor how long bugs stay open, who is working on them, and how quickly they are resolved. Custom filters and dashboards also help QA engineers track specific types of defects, such as critical issues or bugs affecting certain features.



2. Jira Login and User Navigation

Question: How do you log in to Jira and navigate its interface effectively?

Answer: To log in to Jira, follow these steps:

- Open your browser and go to your company's Jira URL, which usually looks something like `https://<company>.atlassian.net`
- Enter your credentials (email and password). If your company uses Single Sign-On (SSO), you may need to use your company credentials.
- After logging in, you will reach your Jira homepage, which usually displays recent projects or tasks you're working on. Navigating Jira's interface:
 - Top Navigation Bar: At the top, you'll see options like Projects, Issues, and Boards. From here, you can navigate to various parts of Jira.
 - Projects: Clicking on this will show you a list of all your assigned projects. You can select any project to view its details, including backlogs, sprints, and boards.
 - Issues: This section allows you to view and manage all the issues (user stories, tasks, bugs) assigned to you or your team. You can also create new issues from here.
 - Boards: This leads you to Agile boards (Scrum or Kanban), where you can see the flow of tasks, from to-do to done.

For example, if you're working in a sprint, you can navigate to the Boards section, select the sprint board, and drag user stories or tasks through their respective stages (To Do, In Progress, Done). The sidebar on the left may have shortcuts to recently viewed issues or boards, allowing you to quickly jump between tasks.

Question: What are the different user roles in Jira? How do permissions work in Jira?

Answer: Jira organizes users into roles, with each role having specific permissions and responsibilities. The main roles you may see are:

- **Jira Administrator:** The admin has full control over the system. He/she manages user accounts, project settings, and permissions. The Administrator can also customize workflows, create new projects, and set up security levels. Example: An admin can configure who has the right to transition an issue from "In Progress" to "Done."
- **Project Administrator:** A project admin manages a specific project. He/she can edit project settings like workflows and permissions within their project but don't have system-wide control. This role can perform project management without having full access to Jira settings. Example: A project admin can create custom fields for their project or modify the project's board layout.
- **Developer/Engineer:** Developers work on tasks or issues assigned to them. They can change issue statuses, comment on tasks, and log time spent on them. They cannot change project settings but have control over their tasks and issues. Example: A developer marks a task as "In Progress" once they start working on it.
- **QA/Test Engineer:** QA engineers can create, view, and update issues. They may have additional permissions to log bugs, execute tests, and track testing progress. They do not have access to project settings but can communicate with developers and project managers on issue statuses. Example: A QA engineer logs a bug, assigns it to a developer, and tracks its resolution throughout the sprint.
- **Viewer/User:** These are users who can view issues and tasks but cannot edit them. This role is often assigned to stakeholders or managers who need to monitor progress without making changes. Example: A product manager can view the progress of a sprint without making any direct updates to tasks.

Question: How is access control managed in Jira?

Answer: Access control in Jira is managed through permission schemes. These schemes define what actions users can take based on their roles. Permissions can be customized for each project, allowing admins to control who can:

- Create, edit, or delete issues
- Transition issues from one status to another
- Access reports, dashboards, or project boards

For example, only developers might have the permission to close a bug, while QA engineers can only move it to "Tested." The level of access is set up so that users can perform their necessary tasks without altering critical project settings.

In addition, Jira supports group permissions, where a set of users is grouped together (e.g. developers, testers), and permissions are assigned to the group as a whole. This simplifies managing permissions across multiple users and projects.

KASPER
ANALYTICS

3. Jira Tickets: Creating and Managing Issues

Question: What is a Jira ticket, and how do you create it?

Answer: A Jira ticket represents any task, bug, issue, or user story that is tracked within the Jira project management system. For example, the Jira ticket can be a bug that needs fixing or a feature that needs development.

Steps to Create a Jira Ticket:

- After logging into Jira, navigate to your project dashboard.
- In the top menu, click on the Create button.
- A ticket creation form will appear. Select the Issue Type (e.g. Bug, Task, Story, Epic) based on what you're reporting.
- Fill in the Summary, Description, and other relevant fields (explained in detail below).
- Assign the issue to the appropriate team member using the Assignee field. You can leave this unassigned if you want someone else to pick it up later.
- Set the Priority to indicate the importance of the ticket (e.g. High, Medium, Low).
- Click Create to save the ticket, and it will appear in your project backlog or active sprint.

Question: What are the key fields of a Jira ticket, and what do they mean?

Answer: When creating or managing a Jira ticket, several important fields help to define and track the issue:

- **Summary:** This is a brief, one-line description of the issue or task. It should be concise but clear and enough to give anyone looking at the ticket a quick idea of what it's about. Example: "Fix the login issue on the customer portal."
- **Description:** This is a detailed explanation of the issue or task. Include steps to reproduce the problem, expected behavior, or specific requirements if it's a task. This field is used for providing context to whoever is working on the

ticket. Example: "When users enter incorrect login credentials, the system crashes instead of showing an error message."

- **Priority:** Priority helps the user understand how urgent the task is. Jira allows you to set levels like High, Medium, Low, or Blocker. Example: A "Blocker" priority might be given to a bug that prevents users from accessing the system.
- **Assignee:** The assignee is the person responsible for working on and resolving the issue. Assigning the correct team member enables accountability and makes the workflow smooth.
- **Status:** This shows the current state of the ticket. Common statuses include To Do, In Progress, and Done. But teams can customize these based on their workflow.
- **Issue Type:** Jira issues can be classified as Bug, Task, Story, Epic, or Sub-task. Each type defines the nature of the issue and how it should be addressed.
- **Attachments:** You can attach files like screenshots, logs, or documents to the ticket. This is useful when additional context is needed, especially for bugs.
- **Comments:** The comment section allows team members to discuss the issue, leave feedback, or ask for clarification. This is needed for collaboration in Jira.

Question: How do you track and update Jira tickets throughout their lifecycle?

Answer: Tracking and updating Jira tickets effectively is needed for managing workflows and keeping the project on track.

- **Updating Ticket Status:** Each Jira ticket moves through various statuses as work progresses. Example: When a developer begins work on a task, he/she moves it from To Do to In Progress. Once the task is complete, he/she updates the status to Done. If it's a bug that needs validation by the QA team, the status might be changed to Ready for Testing.
- **Adding Comments:** Regularly update the ticket's Comments section to provide progress updates or communicate with other team members.

Example: A developer might add a comment saying, "Fixed the issue in the login module. Please test."

- **Time Tracking:** Jira has a built-in Time Tracking feature that allows team members to log the time spent on each ticket. This helps in measuring the effort required to resolve issues and can provide input data for sprint planning.
- **Sub-tasks and Linking Issues:** For larger tasks or user stories, you can break them down into smaller, more manageable sub-tasks. Additionally, if a ticket is related to or dependent on another issue, you can use the Linked Issues feature to show this relationship. Example: If fixing a bug depends on another feature being implemented, you can link the two tickets to indicate that one can't proceed without the other.
- **Using Jira Filters:** Jira provides a robust filtering system to help you track issues. You can create custom filters to see only the tickets that are relevant to you, such as "My Open Bugs" or "All Critical Issues in Sprint."
- **Dashboards and Reports:** Jira's dashboard feature allows you to view your ticket tracking. You can create custom dashboards to show information that is important for you, such as the number of tickets by status, priority, or assignee.

KASPER
ANALYTICS

4. Jira Bug Tracking and Issue Tracking

Question: How do you raise and manage bugs in Jira?

Answer: Raising a bug in Jira is a straightforward process but needs details:

Creating a Bug:

- Click the Create button in Jira.
- Choose Bug from the list of issue types.
- Fill out the Summary (a brief title of the bug). Example: "Login page crashes after entering valid credentials."
- Enter the Description (detailed steps to reproduce the bug). Example:
 - Navigate to the login page.
 - Enter valid username and password.
 - Click login.
 - The page crashes instead of logging in the user.
- Preferably attach supporting information like screenshots or log files.
- Set the Priority (e.g. High, Medium, Low) to indicate the urgency of the bug resolution.
- Assign the bug to a team member responsible for fixing it.

Managing Bugs:

- Once created, bugs can be tracked throughout the software project lifecycle.
- You can update the bug by adding comments, changing the assignee, or modifying fields like status or priority.
- Bugs can be updated with statuses like Open, In Progress, In Review, and Closed to show their current stage.

Question: What is the bug lifecycle in Jira, and how is an issue resolved?

Answer: The bug lifecycle in Jira represents the various stages a bug goes through from when it's raised until it's resolved.

Bug Lifecycle Stages:

- Open: The bug has been logged but not yet assigned to a developer.
- In Progress: A developer is working on fixing the bug.
- In Review: The bug fix is complete and is awaiting review or re-testing by the QA team.
- Resolved: The bug has been fixed and tested, but verification may be needed.
- Closed: The bug is completely resolved, and no further action is needed.
- Reopened (optional): If the issue reappears or was not fixed correctly, it can be reopened and re-evaluated.

Issue Resolution:

- When a developer fixes a bug, the status changes from In Progress to In Review.
- The QA team then tests the bug fix to confirm it's resolved.
- If the fix works as expected, the bug is marked Closed.
- If the issue still persists, the bug is Reopened, and the process starts over.

5. Jira for Agile Project Management

Question: How do you use Jira for managing Agile projects, especially Epics, Stories, and Tasks?

Answer: Jira is popular for Agile project management because it supports Scrum and Kanban methodologies through a customizable interface. Here's how Jira can be used to manage Agile projects:

Epics, Stories, and Tasks:

- **Epics:** These are large user stories or features that can be broken down into smaller, more manageable units. An epic represents a high-level goal, that may need multiple sprints or iterations to be achieved. Example: "Improve User Authentication"
- **Stories:** User stories are smaller, specific features or functionalities that can be completed within a sprint. Stories typically represent a user need or functionality requirement. Example: "As a user, I want to log in using social media credentials."
- **Tasks:** These are actionable items that need to be done to complete a story. Tasks are usually technical or testing work associated with implementing a user story. Example: "Implement OAuth for login."

Managing Agile in Jira:

- In Jira, you can create Epics and then link related Stories and Tasks under each epic.
- Agile boards in Jira (Scrum or Kanban) visually display the progress of epics, stories, and tasks, making it easy to track them.

Question: How do Agile workflows work in Jira?

Answer: Agile workflows in Jira help track the progress of tasks, stories, and epics across different stages of the software development lifecycle. Workflows can be customized to suit the needs of the team but typically follow this structure:

- **Backlog:** All planned stories, tasks, and epics are added to the backlog, representing work that needs to be done in the future. During sprint planning, items from the backlog are moved into the sprint.
- **To Do:** When a sprint starts, tasks and stories move into the To Do column, meaning that they are ready to be worked on.
- **In Progress:** Once a team member starts working on a task, it moves to the In Progress column. This helps the team track work that is actively being developed.
- **In Review/Testing:** After development, tasks move to In Review or Testing, where they are tested by QA or reviewed by peers before moving to completion.
- **Done:** Once a task, story, or epic is fully developed, tested, and accepted, it is moved to Done, marking it as completed.

Example Workflow: A user story "Implement social media login" starts in the backlog.

Once the sprint begins, it moves to To Do. After a developer starts work, it's marked as In

Progress, followed by In Review when the implementation is complete. Once QA tests the feature, it moves to Done.

6. Jira Sprint Management

Question: How do you start and manage sprints in Jira?

Answer: To start and manage sprints in Jira, you might follow these steps:

Create a Sprint: On your Scrum board, you'll find the option to create a sprint in the backlog view. Click on "Create Sprint." Add stories or tasks from the backlog to the sprint by dragging them into the sprint section.

Plan the Sprint: During the sprint planning meeting, the team decides which stories or tasks will be completed in the upcoming sprint. You can assign tasks to team members and estimate their story points (effort estimation). You may want to use my Agile Test Estimator tool for planning your sprint.

Start the Sprint: Once the sprint plan is ready, click the Start Sprint button. This will lock the selected stories into the sprint and start tracking their progress. A sprint usually lasts for 1-4 weeks, depending on the team's preferences.

Managing the Sprint: The Scrum board displays the sprint's tasks in columns like To Do, In Progress, and Done. Move tasks across these columns as they progress. Jira also has the feature to add comments, log work, and track any blockers during the sprint.

Question: What is the sprint backlog, burndown charts, and velocity tracking in Jira?

Answer: **Sprint Backlog:** The sprint backlog contains all the user stories and tasks committed to the sprint. It's a prioritized list of work to be done during the sprint. Jira's Backlog view makes it easy to organize and manage items by dragging them into the sprint, with details like story points and task assignments available.

Burndown Chart: A burndown chart shows the team's progress throughout the sprint. It shows how much work remains and whether the team is on track to complete the sprint on time. Jira automatically generates this chart, updating it as tasks move to the Done column. The x-axis represents time (days in the sprint), and the y-axis represents remaining work (story points or tasks). A sharp decline

means the team is finishing tasks, while a flat line indicates a lack of progress.

Example: A burndown chart might show that by the

middle of the sprint, half of the tasks are completed, indicating good progress. If it's flat near the end, it suggests a risk of incomplete work.

Velocity Tracking: Velocity means the amount of work the team completes in each sprint, typically measured in story points. Jira tracks velocity by comparing completed story points across sprints, helping the team understand its capacity and estimate future sprints more accurately. Teams should try to maintain consistent velocity. If velocity fluctuates, it could indicate issues with task estimation or unforeseen challenges during the sprint. Example: If a team completes 30 story points in Sprint 1 and 28 story points in Sprint 2, the velocity is fairly consistent. This data helps the team predict that they can complete around 30 story points in the next sprint.

KASPER
ANALYTICS

7. Jira Reporting and Metrics for Testers

Question: How do you generate reports in Jira for sprint progress, defect density, and test coverage?

Answer: Jira provides built-in reports that can be used to track progress and quality metrics in software projects. Some reports that are useful for testers include:

- **Sprint Reports:** A Sprint Report helps track the progress of a sprint. It shows what was completed, what wasn't, and the work that was carried over to the next sprint. To generate this report, go to your project, click on Reports in the sidebar, and select Sprint Report. Then choose the sprint you want to analyze, and Jira will show you a summary of the sprint's performance.
- **Defect Density Report:** Defect density measures the number of defects found in a module or set of features relative to the size of the code or effort (typically defects per 1000 lines of code). You can create a custom "Defect Density" report to track defects based on the module or release. Then use Filters to search for issues categorized as defects, and sort them by component or release to calculate defect density. You can also export this data to external tools for more detailed calculations.
- **Test Coverage Report:** Jira integrates with plugins like Zephyr or Xray to generate detailed test coverage reports. These tools allow you to see which requirements or user stories have been covered by test cases. To generate a test coverage report in Zephyr, navigate to Test Reports and choose the coverage report. Then, it will display the percentage of test cases executed for each requirement.

Question: How can you set up Jira dashboards for QA metrics?

Answer: Setting up a Jira dashboard allows testers to monitor key metrics like bug trends, test progress, and overall project health. Here's how to create a dashboard with QA- specific metrics:

1. Create a New Dashboard:

- Click on Dashboards in the Jira menu and select Create Dashboard.
- Name your dashboard and set the sharing preferences (private, project-specific, or global access for the team).

2. Add Gadgets for QA Metrics:

- Jira provides several gadgets that can be added to your dashboard to visualize different metrics:
- Issue Statistics Gadget: Useful for showing bug trends or test execution progress. You can configure it to display issue types (e.g. bugs, tasks) by priority or status.
- Two-Dimensional Filter Gadget: This allows you to compare multiple dimensions, like bug severity vs. assignee, or test case status vs. priority.
- Test Execution Gadget (Zephyr or Xray): If you're using a test management plugin like Zephyr, this gadget shows test execution progress and pass/fail rates for test cases.

3. Monitor the Key Metrics:

- Once your dashboard is set up, it will automatically refresh with data. For example:
- Bug trends: Track the number of new vs. resolved bugs over time.
- Test case progress: View how many test cases are passed, failed, or still in progress.
- Cycle time: Track how long it takes to resolve bugs and complete testing tasks, helping identify bottlenecks.

With these dashboards, QA engineers can keep track of the project's quality and share it with the team during daily stand-ups or sprint reviews.

8. Jira Automation Integration with Testing Tools

Question: How can Jira be integrated with test automation frameworks like Selenium and Jenkins?

Answer: Integrating Jira with test automation frameworks such as Selenium and Jenkins allows teams to automate the testing process and link the results directly to Jira issues. This helps in tracking automated test results and associating them with relevant bugs or features. Here's how to set up these integrations:

- **Selenium Integration:** Selenium is a popular tool for automating browser-based tests. By integrating it with Jira, testers can automatically create Jira tickets for test failures and track test results. Typically, this integration happens through third-party plugins or custom scripts. Example 1: Zephyr is a Jira plugin that connects with Selenium. You can configure Zephyr to automatically push the test results from Selenium to Jira. Example 2: Custom API: Using Jira's REST API, you can create a script that automatically updates Jira issues with Selenium test results, including details like screenshots of failures or logs.
- **Jenkins Integration:** Jenkins is a popular CI/CD tool used to automate build and deployment pipelines. Integrating Jenkins with Jira allows the team to link build status and test results with Jira issues. To integrate Jenkins with Jira:
 - Use the Jira Plugin for Jenkins, which allows Jenkins jobs to update Jira issues with build information.
 - Configure Jenkins to automatically update the status of Jira tickets (e.g. close an issue once the build passes or transition a ticket to "In Progress" when a build starts).
 - Jenkins can also push automated test results back to Jira, linking them with the corresponding issues or stories.

Question: How can automated test results be linked to Jira tickets?

Answer: Linking automated test results to Jira tickets is needed for establishing traceability between test automation and test results. There are multiple ways to do this:

1. Using Test Management Plugins: Tools like Zephyr or Xray can be used for managing test cases within Jira. They allow you to link automated test cases directly to Jira tickets (e.g. user stories or bug reports). Once a test case runs, the result (pass or fail) is automatically updated in Jira and associated with any related issue.
2. Jira REST API: If you're using a custom automation framework, you can use the Jira REST API to update Jira issues based on test results. Example: after a test run, the framework can send a request to Jira's API to update an issue with the results:
 - Attach logs or screenshots of failed tests.
 - Transition the issue's status (e.g. from "In Progress" to "Ready for Testing").
 - Comment directly on the Jira ticket with details from the test run.
3. Continuous Integration Tools: When using CI/CD tools like Jenkins or Bamboo, test results can be linked to Jira issues automatically. These tools can be set up to trigger tests and report the outcome back to Jira once the build process completes. Jenkins, for example, can automatically link build jobs and test results to Jira tickets. If a build fails, the relevant Jira issue is updated, and if a build succeeds, the issue status is automatically transitioned to the next phase.

9. Customizing Jira Workflows

Question: How can Jira workflows be customized for different projects?

Answer: A Jira workflow means the series of steps an issue or task goes through, from

creation to completion. Customizing workflows allows teams to tailor Jira's processes to

fit their project requirements.

A Jira workflow is made up of statuses, transitions, and resolutions:

- Statuses represent the current state of an issue (e.g. "To Do," "In Progress," "Done").
- Transitions are the paths that move an issue from one status to another (e.g. moving from "In Progress" to "Review").
- Resolutions define the final state of an issue (e.g. "Fixed," "Won't Fix").

Customizing Workflows: Jira offers a standard workflow, but you can customize it to

better suit your project's needs. For example, a QA team might need a separate "Testing" status in the workflow. To customize a workflow:

- Navigate to Jira Settings > Issues > Workflows.
- Select an existing workflow or create a new one.
- You can add, edit, or delete statuses. For instance, you can add statuses like "Under Review" or "Ready for QA" to track issue progress more accurately.
- Define transitions between statuses. You can control the direction and conditions of these transitions (e.g. only users in the "QA" group can move an issue to "Testing").
- Save and publish the workflow once changes are made.

Question: What are workflow transitions and permissions in Jira?

Answer: Workflow transitions in Jira define how an issue moves from one status to another. For example, an issue can transition from "In Progress" to "Testing" when development is complete.

Setting Up Transitions: When customizing a workflow, you can specify which transitions are allowed between statuses. Each transition can have conditions, validators, and post- functions attached to it:

- Conditions restrict who can perform the transition. For example, only users with the "Developer" role might be able to move an issue from "To Do" to "In Progress."
- Validators ensure that certain criteria are met before a transition can occur (e.g. requiring all mandatory fields to be filled out).
- Post-functions automate actions that should happen after a transition, such as assigning the issue to a different user or sending a notification.

Managing Permissions: Permissions in Jira control who can view, edit, or transition issues. These can be configured to match your project's needs and team roles. You can control:

- Who can transition issues: For example, only members of the QA team can move issues to "Testing."
- Who can edit specific fields during a transition: Only the assignee might be allowed to update the description when moving an issue to "In Progress."

Permissions are managed via Permission Schemes, which are assigned to different projects. Within the scheme, you can configure who has access to specific actions (like transitioning an issue or closing it) based on user roles (e.g. Developers, Testers, Admins). Example: In an Agile project, only users with the "Product Owner" role can transition an issue from "Ready for Deployment" to "Deployed." This means that only authorized users can approve final releases to production.

10. Defect Lifecycle Management in Jira

Question: What are the steps to raise, assign, fix, and close defects in Jira?

Answer: Managing defects in Jira involves a structured process to ensure that bugs are logged, tracked, and resolved efficiently.

Raising a Defect: To log a defect in Jira, you can take the following steps:

- Click on Create to open a new issue.
- Select Bug as the issue type.
- Fill in the fields:
 - Summary: A concise description of the defect.
 - Description: Detailed steps to reproduce the defect, the expected behavior, and the actual behavior.
 - Priority: Assign a priority based on the severity of the defect (e.g. Blocker, Critical, Major, Minor).
 - Assignee: Assign the defect to the appropriate developer or team member.
 - Attachments: Add any screenshots, logs, or supporting documentation.
- Click Create to submit the defect.

Assigning the Defect: Once the defect is logged, it needs to be assigned to the responsible developer or team member. You can do this by either setting the Assignee field during creation or updating it after. The assignee can then move the defect to the next status (e.g., "In Progress" or "Open for Review").

Fixing the Defect: When the developer picks up the defect, they will change its status to In Progress to indicate that they are working on resolving it. After the fix is implemented, the developer moves the defect to a status like Ready for Testing or Resolved, indicating that it is ready for validation by the QA team.

Closing the Defect: The QA team tests the fix. If the issue is resolved, they update the status to Closed. If the defect is not fixed correctly or causes new issues, the defect is reopened, and the cycle repeats.

Question: How are defect statuses managed throughout the sprint?

Answer: In Jira, managing defect statuses during a sprint allows the team to track the progress of bug fixes and prevent unresolved issues from impacting the sprint's success.

Defect Statuses in a Sprint: During a sprint, the lifecycle of a defect typically follows these statuses:

- Open: The defect has been raised and needs to be reviewed.
- In Progress: The developer is working on resolving the defect.
- Ready for Testing: The defect is fixed and ready for testing by the QA team.
- Closed: The QA team has validated the fix, and the defect is resolved.
- Reopened: If the defect persists after the fix, it can be reopened for further investigation.

Tracking Defect Progress: As part of the Sprint Board, defects appear in various columns depending on their status (e.g. "To Do," "In Progress," "Done"). This makes it easy for the team to see the current state of each defect. The Burndown Chart helps track the remaining effort related to defects and other tasks, ensuring the team is on track to complete the sprint goals.

Closing Defects Before Sprint Completion: All defects should ideally be resolved before the sprint ends. If a defect cannot be fixed within the sprint, it should be discussed during the sprint review, and the team can decide whether to carry it over to the next sprint.

These questions and answers are only based on my understanding. Please visit the Atlassian official website for the latest and most accurate information. The Jira link is <https://www.atlassian.com/software/jira>


Need any experience/support on
Hands-on Live Project
and Live Frameworks.

Please
Connect



 Contact Number - 8130877931

 Email - hr@kasperanalytics.com

 LinkedIn - www.linkedin.com/company/kasper-analytics/