Jira와 GitHub 자동화 통합

개발 워크플로우 전 과정을 Jira와 GitHub 간 양방향 자동화 파이프라인으로 통합

Jira

이슈 추적 및 프로젝트 관리

GitHub

코드 저장소 및 CI/CD

양방향 자동화

이슈 자동 생성

브랜치 자동 매핑

상태 자동 동기화

프로젝트 개요 및 목표

프로젝트 개요

본 프로젝트는 Jira와 GitHub를 연동하여 개발 워크플로우 전반을 자동화하는 것을 목표로 합니다. 이는 기획부터 개발 완료에 이르는 모든 과정을 통합하여, 개발 팀의 효율성을 극대화하고 프로젝트 관리의 투명성을 확보하기 위한 것입니다.

핵심 목표

개발 활동의 가시성 향상

GitHub의 브랜치, 커밋, 풀 리퀘스트를 Jira 작업 항목과 연결하여 개발 진행 상황을 한 곳에서 모니터링

수동 업데이트 감소

개발 활동에 따른 Jira 상태 변경을 자동화하여 수작업으로 인한 오류와 시간 낭비 줄이기

협업 강화

개발자와 프로젝트 관리자가 각자의 도구에서 작업하면서도 정보를 공유할 수 있는 환경 제공

CI/CD 프로세스 지원

GitHub의 빌드와 배포를 Jira와 연결하여 지속적인 통합 및 배포 프로세스 지원

보안 취약점 관리

GitHub Advanced Security에서 감지된 취약점을 Jira에서 추적하고 관리할 수 있도록

릴리스 관리 자동화

GitHub에서 버전이 생성되면 Jira에서 해당 버전을 자동으로 릴리스 처리

핵심 구성요소

이 자동화 파이프라인을 구축하기 위한 핵심 기술 구성요소입니다.

Jira Automation (이슈 이벤트 트리거)

Jira 내에서 발생하는 다양한 이슈 이벤트(생성, 수정, 상태 변경 등)를 감지하고, 이에 반응하여 미리 정의된 자동화 규칙을 실행하는 역할을 합니다.

이벤트 감지 → 자동화 실행

GitHub Actions (repository_dispatch)

GitHub 저장소에서 특정 이벤트를 수신하거나, 외부 시스템으로부터 repository_dispatch 이벤트를 받아 워크플로우를 실행하는 CI/CD 플랫폼입니다.

외부 요청 → GitHub 워크플로우 실행

REST API 기반 데이터 연동

Jira와 GitHub 간의 데이터 교환은 각 플랫폼이 제공하는 RESTful API를 통해 이루어집니다. 이를 통해 이슈 정보, 코드 변경 사항, 상태 업데이트 등 다양한 데이터를 양방향으로 주고받습니다.

양방향 데이터 교환

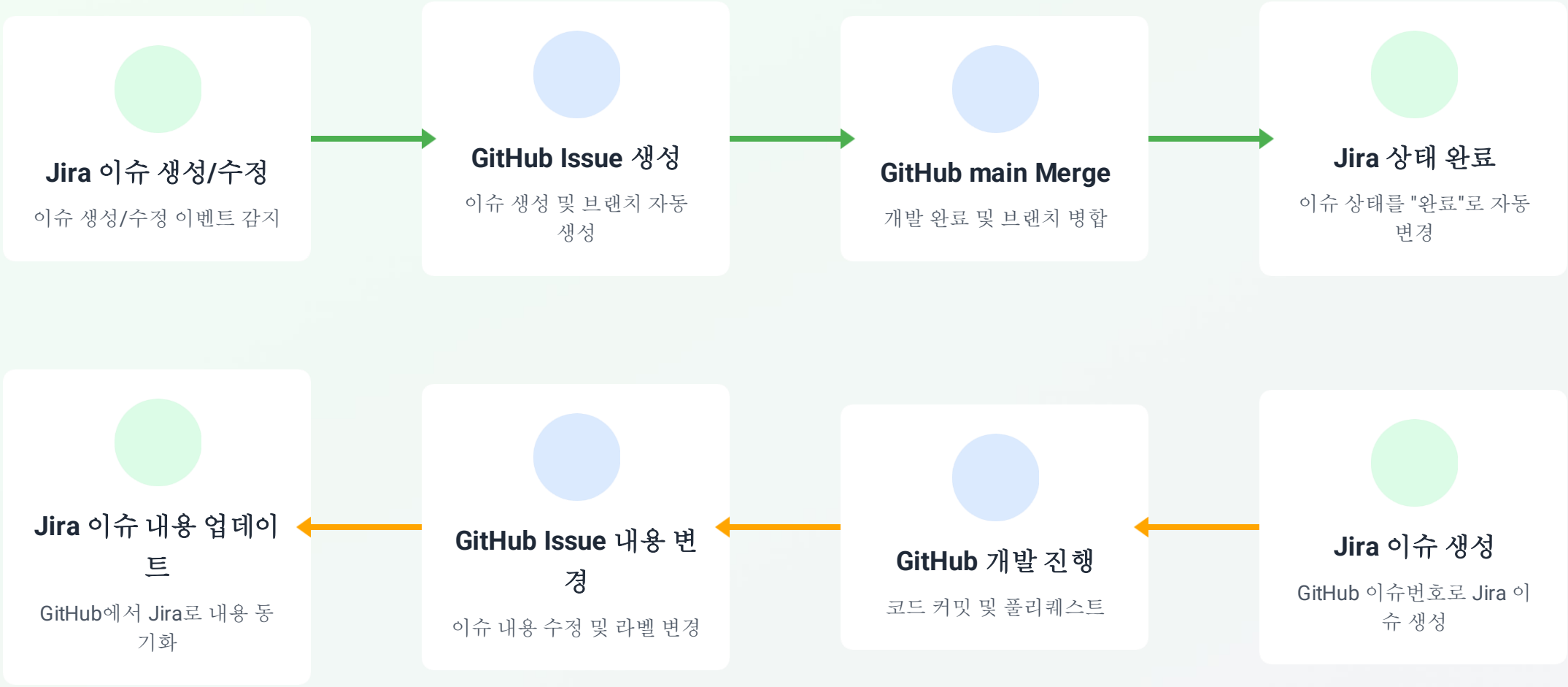
커스텀 필드로 이슈 매핑

Jira 이슈와 GitHub 이슈를 정확하게 연결하기 위해 Jira 내에 'GitHub Issue Number'와 같은 커스텀 필드를 활용합니다. 이 필드에 GitHub 이슈 번호를 저장하여 두 시스템 간의 고유한 매핑 관계를 유지합니다.

고유 매핑 관계 유지

전체 자동화 흐름도

Jira와 GitHub 간의 양방향 자동화 파이프라인 흐름



Jira → GitHub: 이슈 생성/수정 → Issue/Branch 생성 → main Merge → 상태 완료

GitHub → Jira: Issue 내용 변경 → GitHub 이슈 수정 → 개발 진행 → 이슈 생성

Jira → GitHub: 이슈 및 브랜치 자동 생성

Jira 이슈 생성
Story / Task / Bug

repository_dispatch
GitHub Actions 호출

GitHub Issue 생성
자동 브랜치 생성

트리거 및 워크플로우

트리거: Jira 이슈 생성
Jira에서 'Story', 'Task', 'Bug' 유형의 이슈가 생성될 때 자동화 시작

워크플로우 파일
.github/workflows/jira-issue-created.yml

repository_dispatch
Jira Automation이 GitHub Actions 워크플로우를 호출하는 이벤트

주요 기능

이슈 본문 생성
Jira 이슈 유형별로 미리 정의된 Markdown 템플릿으로 GitHub 이슈 본문 생성

라벨 자동 지정
Jira 이슈 유형에 따라 GitHub 라벨 자동 부여 (Story→Story, Bug→Bug, Task→Jira Label)

브랜치 자동 생성
Jira 이슈 키를 포함하는 명명 규칙에 따라 개발 브랜치 자동 생성 (ex: feature/SCRUM-123)

생성 후 매핑
생성된 GitHub 이슈의 고유 번호는 Jira의 커스텀 필드(customfield_10058)에 저장되어 이후 업데이트 자동화에서 참조됨

자동화된 이슈 생성 워크플로우

Jira Automation
이슈 생성/수정 감지

GitHub Actions
Workflow 실행

Issue 생성
GitHub에 이슈 생성

1

트리거 이벤트

Jira에서 새로운 'Story', 'Task', 또는 'Bug' 유형의 이슈가 생성될 때, Jira Automation이 repository_dispatch 이벤트를 트리거합니다.

```
repository_dispatch: .github/workflows/jira-issue-created.yml
```

2

템플릿 선택

이슈 유형에 따라 적절한 Markdown 템플릿을 선택합니다.

- Story: automation-story.md
- Bug: automation-bug.md
- Task: automation-task.md

3

이슈 내용 생성

선택된 템플릿을 기반으로 GitHub 이슈의 본문을 생성합니다. 이는 GitHub 이슈의 내용이 일관된 형식과 구조를 갖도록 보장합니다.

```
## 📄 이슈 개요 {description} --- ### 🧑‍💻 생성자 {initiator} ...
```

4

라벨 자동 지정

생성되는 GitHub 이슈에는 Jira 이슈 유형에 따라 적절한 라벨이 자동으로 부여됩니다.

- Story → Story 라벨
- Bug → Bug 라벨
- Task → Jira의 Label 값(Feature / Refactor / Documentation)

라벨 자동 지정 및 브랜치 생성

라벨 자동 지정

Jira 이슈 유형에 따라 GitHub에 적절한 라벨이 자동으로 부여됩니다.

Story 유형 — GitHub: Story

Bug 유형 — GitHub: Bug

Task 유형 — GitHub: Feature/Refactor/Documentation

Jira의 Label 값에 따라 GitHub 라벨이 지정됩니다.

브랜치 자동 생성

Jira 이슈 키를 포함하는 명명 규칙에 따라 개발 브랜치가 자동으로 생성됩니다.

Jira 이슈 키

SCRUM-123

자동 생성된 브랜치

feature/SCRUM-123

개발자가 수동으로 브랜치 생성하는 번거로움을 줄이고 명명 규칙의 일관성을 유지합니다.

브랜치 자동 생성의 이점

- 개발 속도 향상
- 명명 규칙 일관성
- Jira-GitHub 연동
- 팀원 협업 용이

Jira → GitHub: 이슈 내용 업데이트 자동화

트리거

Jira 이슈 수정 (Description / Priority / Labels / Duedate 등)

```
.github/workflows/jira-issue-updated.yml
```

업데이트 프로세스

GitHub 이슈 식별
customfield_10058

내용 즉시 갱신
Jira → GitHub

Markdown 재렌더링
구조 일관성 유지

오류 처리 및 재시도

404 오류 발생 시 제목·본문 기반으로 대상 Issue 자동 탐색 후 재시도

주요 기능

이슈 매핑

customfield_10058 값을 사용하여 Jira 이슈와 GitHub 이슈를 정확히 연결하고, 이 매핑 정보는 이후 업데이트 자동화에서 참조됩니다.

Markdown 템플릿

이슈 본문은 미리 정의된 Markdown 형식으로 재렌더링되어 일관된 구조와 포맷을 유지합니다.

```
## 📄 이슈 개요 {{description}} --- ### 🧑‍💻 생성자 {{initiator}}
```

이슈 업데이트 기능 및 오류 처리

이슈 업데이트 프로세스

GitHub 이슈 식별

Jira 이슈의customfield_10058 값을 사용

내용 즉시 갱신

변경된 Jira 정보로이슈 본문/제목/라벨 업데이트

마크다운 재렌더링

이슈 본문을Markdown 형식으로 재렌더링

Markdown 템플릿 예시:

```
## 📄 이슈 개요 {{description}} -- ### 🧑‍💻 생성자 {{initiator}}
```

오류 처리 및 재시도 메커니즘

오류 감지

GitHub 이슈를 찾지 못해 404 오류 발생

자동 탐색

이슈 제목이나 본문 내용을 기반으로 대상 이슈 자동 검색

업데이트 재시도

검색된 이슈로 업데이트를 재시도하여 데이터 불일치 최소화

자동화 이점: Jira에서 수정된 내용이 GitHub 이슈에 자동으로 반영되도록 하여, 개발자가 항상 최신 정보를 확인할 수 있도록 돕고, 이슈 본문의 구조와 포맷 일관성을 유지합니다.

GitHub → Jira: 상태 및 내용 자동 동기화

내용 변경 동기화

트리거:

GitHub Issue 내용 수정 (Description / Priority / Labels)

처리 과정:

GitHub Actions 워크플로우는

GitHub → Jira 기술 구현 포인트

GitHub 이벤트 활용

issues.edited 이벤트
GitHub 이슈 내용 변경 시 자동화 트리거

push 이벤트
브랜치 병합 시점에 자동화 트리거

Jira API 호출 방식

이슈 내용 업데이트

```
/rest/api/3/issue/{key}
```

이슈 상태 전환

```
/rest/api/3/issue/{key}/transitions
```

브랜치명 파싱

feature/SCRUM-123 같은 브랜치명에서 Jira 이슈 키 추출

정규 표현식 활용

```
const issueKey = branchName.match(/([A-Z]+-[0-9]+)/)[0];
```

feature/SCRUM-123 **SCRUM-123**

GitHub Secrets

Jira API 호출에 필요한 인증 정보 안전히 보관

JIRA_BASE_URL
https://your-jira-domain.com

JIRA_USER_EMAIL
user@example.com

JIRA_API_TOKEN
API 토큰 (보안상의 이유로 GitHub Secrets에 저장)

민감한 정보의 노출을 방지하고 보안을 강화

구현 현황 및 향후 계획

구현 진도

Jira → GitHub

이슈 및 브랜치 자동 생성

이슈 내용 업데이트

GitHub → Jira

이슈 내용 반영 (Description / Priority / Label 동기화)

main Merge 시 Jira 자동 완료 처리

향후 확장 계획

Slack 알림 연동

이슈 생성 및 완료 시점에 Slack 채널에 자동으로 알림을 전송하여 팀원들이 중요한 변경 사항을 실시간으로 인지하고 빠르게 대응할 수 있도록 지원

정보 확산 속도 향상 및 수동 확인 절차 줄이기

다음 개선 방향

더 많은 이슈 유형 지원

자동화 속도 향상

오류 처리 및 재시도 메커니즘 개선

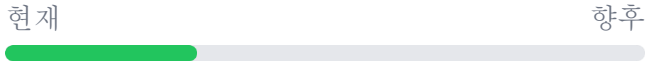
다른 도구들과의 통합 확장

기대 효과

이 자동화 파이프라인의 도입은 개발 프로세스 전반에 걸쳐 다음과 같은 핵심적인 긍정적 효과를 가져올 것으로 기대됩니다.

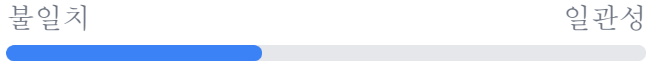
이슈 관리 완전 자동화

개발 워크플로우의 시작부터 끝까지 수동적인 이슈 관리 작업을 최소화하여, 개발 팀이 핵심 업무에 집중할 수 있도록 지원합니다.



Jira와 GitHub 간 데이터 일관성 확보

양방향 동기화를 통해 두 플랫폼 간의 정보가 항상 최신 상태로 유지되어, 정보의 신뢰성을 높이고 혼란을 방지합니다.



개발-운영 간 협업 효율 극대화

개발자와 프로젝트 관리자가 각자의 주력 도구에서 작업하면서도 실시간으로 정보를 공유하고, 투명한 진행 상황을 파악할 수 있게 하여 협업의 효율성을 증대시킵니다.

추가적인 효과

- GitHub 이슈의 제목이나 본문 내용을 기반으로 대상 이슈를 자동 탐색하여 업데이트를 재시도합니다.
- Jira에서 수정된 내용이 GitHub 이슈에 자동으로 반영되어, 개발자가 항상 최신 정보를 확인할 수 있습니다.

이 자동화는 개발 팀의 생산성을 크게 향상시키며, 팀 간의 소통을 원활하게 만들어 프로젝트의 성공적인 수행을 보장합니다.