記事
Mihoko Iijima · 2020年4月28日 2m read

GitLabを使用したInterSystemsソリューションの継続的デリバリー - 索引

この連載記事では、InterSystemsの技術とGitLabを使用したソフトウェア開発に向けて実現可能な複数の手法を紹介し、議論したいと思います。 次のようなトピックについて説明します。

- 第1回
  - Gitの基本、Gitの概念を高度に理解することが現代のソフトウェア開発にとって重要である理由
  - Gitを使用してソフトウェアを開発する方法（Gitのフロー）
- 第2回
  - GitLabワークフロー -
    アイデアからユーザーフィードバックまでの完全なソフトウェアライフサイクルプロセス。
  - 継続的デリバリー - チームが短いサイクルでソフトウェアを作成し、ソフトウェアをいつでも確実にリリースできるようにするソフトウェアエンジニアリング手法です。 この手法はソフトウェアの構築、テスト、リリースをより速く、より頻繁に行うことを目指しています。
- 第3回
  - GitLabのインストールと構成
  - 利用環境のGitLabへの接続
- 第4回
  - 継続的デリバリーの構成
- 第5回
  - コンテナとその使用方法（および使用する理由）。
- 第6回
  - コンテナを使用した継続的デリバリーパイプラインの主要コンポーネント
  - それらすべての連携方法。
- 第7回
  - コンテナを使用した継続的デリバリーの構成
- 第8回
  - InterSystems Cloud Managerを使用した継続的デリバリーの構成
- 第9回
  - コンテナアーキテクチャ

この連載記事では、継続的デリバリーの一般的な手法について説明しました。 これは非常に広範なトピックであり、この連載記事の内容は完成されたものではなく、レシピを集めたものとして考えてください。 アプリケーションのビルド、テスト、配信を自動化したい場合、一般的には継続的デリバリー、特にGitLabが最適です。

継続的デリバリーとコンテナを使えば、必要に応じてワークフローをカスタマイズできます。

#Git #継続的デリバリー #その他

ソースURL:
https://jp.community.intersystems.com/post/gitlab%E3%82%92%E4%BD%BF%E7%94%A8%E3%81%97%E3%81%9Fintersystems%E3%82%BD%E3%83%AA%E3%83%A5%E3%83%BC%E3%82%B7%E3%83%A7%E3%83%B3%E3%81%AE%E7%B6%99%E7%B6%9A%E7%9A%84%E3%83%87%E3%83%AA%E3%83%90%E3%83%AA%E3%83%BC-%E7%B4%A2%E5%BC%95