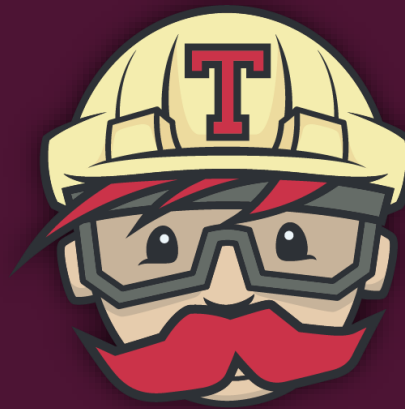

SYSTEMY CI



CZYM JEST CONTINUOUS INTEGRATION?

Ciągła integracja (ang. continuous integration) to praktyka stosowana w trakcie rozwoju oprogramowania, polegająca na częstym, regularnym włączaniu (integracji) bieżących zmian w kodzie do głównego repozytorium i każdorazowej weryfikacji zmian, poprzez zbudowanie projektu (jeśli jest taka potrzeba) i wykonanie testów jednostkowych.

PRACA NAD PROJEKTEM BEZ CIĄGŁEJ INTEGRACJI

Typowe kroki podejmowane w procesie tworzenia oprogramowania:

- Deweloper tworzy nową funkcjonalność bądź wprowadza jakąś łatkę, co oznacza zmianę kodu źródłowego projektu.
- Zmiany zostają dodane do głównego repozytorium projektu (np. w repozytorium GitHub).
- Jeżeli istnieje taka potrzeba kod źródłowy jest kompilowany a następnie wbudowany w wersję do wdrożenia.

PRACA NAD PROJEKTEM BEZ CIĄGŁEJ INTEGRACJI

- Deweloper tworzy nową funkcjonalność bądź wprowadza jakąś łatkę, co oznacza zmianę kodu źródłowego projektu.
 - ↑ Skąd wiadomo, że zmiana wprowadzona przez dewelopera jest możliwa do integracji z projektem?
- Zmiany zostają dodane do głównego repozytorium projektu (np. w repozytorium GitHub).
- Jeżeli istnieje taka potrzeba kod źródłowy jest kompilowany a następnie wbudowany w wersję do wdrożenia.
 - ↑ Kto jest odpowiedzialny za wykonanie dwóch pozostałych kroków?

CI OPIERA SIĘ NA SYSTEMIE KONTROLI WERSJI



GitHub

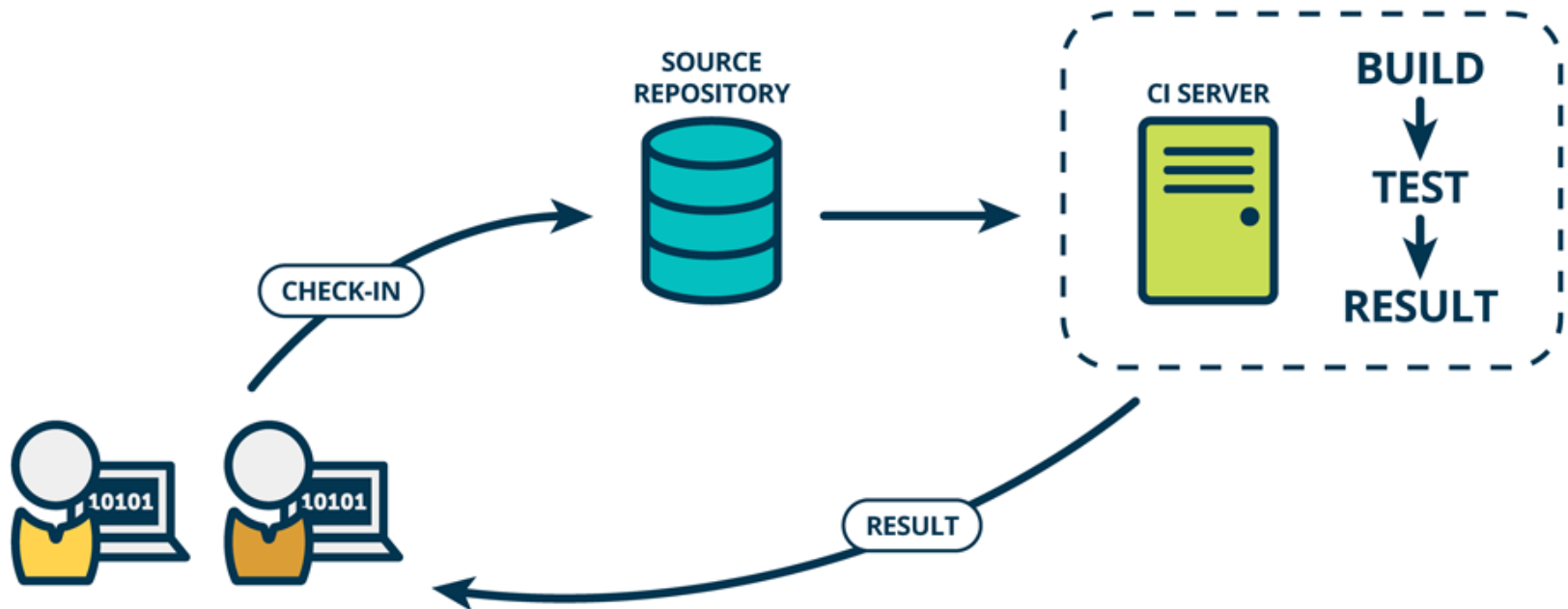
SERWER CIĄGŁEJ INTEGRACJI

Równie istotnym elementem ciągłej integracji jest serwer ciągłej integracji, który musi posiadać dostęp do współdzielonego repozytorium kodu, aby **automatycznie**, bez udziału człowieka, sprawdzić czy każda kolejna zmiana dokonana przez programistę jest integralna z obecną wersją kodu oraz wszystkimi modyfikacjami wprowadzanymi przez pozostałych członków zespołu.

W JAKI SPOSÓB SIĘ TO ODBYWA?

W momencie wysłania przez programistę zmian do repozytorium kodu serwer ciągłej integracji samoczynnie wykrywa i pobiera tę zmianę, buduje system (produkt) oraz **uruchamia testy**. Jeżeli wszystkie testy zakończą się powodzeniem, serwer automatycznie informuje zespół o sukcesie (tzw. zielony build). W sytuacji, kiedy przynajmniej jeden test się nie powiedzie (tzw. czerwony build), serwer automatycznie ostrzega zespół.

Należy tutaj podkreślić, że produkt, którego kod źródłowy nie jest pokryty wystarczającą liczbą dobrych testów, jest kiepskim kandydatem do ciągłej integracji.



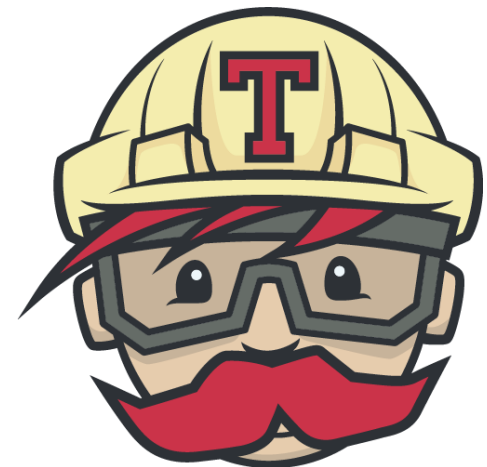
KORZYŚCI PŁYNĄCE Z CIĄGŁEJ INTEGRACJI

- Ułatwia czasochłonny i podatny na błędy proces integracji kodu,
- Zwiększa wydajność zespołu twórczego,
- Przyspiesza wykrywanie potencjalnych błędów.

PRAKTYKA CI – DOBRE PRAKTYKI

- Integrację zaleca się przeprowadzać kilka razy dziennie lub po każdym commitcie,
- Należy posiadać narzędzie do kontroli wersji,
- Należy posiadać narzędzie do budowania (serwer ciągłej integracji),
- Aplikacja powinna być w odpowiednim stopniu pokryta testami,
- Stosowanie pull requestów wraz z ciągłą integracją pomaga nie tylko uniknąć błędów, ale również pozwala utrzymać pewien standard czystości i poprawności kodu.

NAJPOPULARNIEJSZE CI



CIRCLECI



circleci



Funkcje :

CircleCI to system oparty na chmurze - nie jest wymagany serwer dedykowany i nie trzeba go administrować.

Ma darmowy plan nawet dla konta biznesowego

Rest API - masz dostęp do projektów, kompilacji i artefaktów. Rezultatem kompilacji będzie artefakt lub grupa artefaktów. Artefaktami mogą być skompilowane aplikacje lub pliki wykonywalne (np. Android APK) lub metadane (np. Informacje o wynikach testów)

CircleCI jest kompatybilny z:

Python, Node.js, Ruby, Java, Go, itp

Ubuntu (12.04, 14.04), Mac OS X (konto płatne)

Github, Bitbucket

AWS, Azure, Heroku, Docker, serwer dedykowany

Jira, HipChat, Slack



Plusy CircleCI:

- Szybki start

- CircleCI ma bezpłatny plan dla projektów korporacyjnych

- Rozpoczęcie jest łatwe i szybkie

- Lekka, łatwa do odczytania konfiguracja YAML


- Nie potrzebujesz żadnego dedykowanego serwera do uruchamiania CircleCI

Wady CircleCI:

- CircleCI obsługuje tylko 2 wersje Ubuntu za darmo (12.04 i 14.04) oraz MacOS odpłatnie

JENKINS





Jenkins jest jednym z najstarszych projektów open source (2011) wśród narzędzi do ciągłej integracji i mimo to wciąż jest powszechnie używany.

Tak długie życie oprogramowania ma zalety i wady.

Architektura tego narzędzia została dość dobrze już poznana przez jego użytkowników co oznacza, że na jakikolwiek napotkany problem znajdziemy rozwiązanie wśród jego wysoko rozwiniętej społeczności.

Jednak ze względu na dużą bazę 'legacy codu' oraz konieczność kompatybilności wstecznej jego wewnętrzna struktura jest dosyć przestarzała.

Jenkins ma rozbudowany ekosystem wtyczek, który zapewnia wiele nowoczesnych funkcji, jednak są one zazwyczaj opracowywane przez użytkowników i mogą różnić się jakością i niezawodnością.

W ostatnich latach Jenkins opracował tzw. „Pipelines”. Umożliwiają one programistom deklarowanie i opisywanie procesu budowania i wdrażania. Jenkins umożliwia również tworzenie modułów, które można ponownie używać w różnych projektach w celu standaryzacji i usprawnienia wspólnych procesów.

W skrócie, Jenkins ma długą historię rozwoju i użytkowania, dużą i aktywną społeczność i jest wysoce konfigurowalny. Być może z tych powodów „nikt nie został zwolniony za wybór Jenkinsa”.

\$69

PER MONTH

Bootstrap

IDEAL FOR HOBBY PROJECTS

1

 Concurrent job

- ✓ Unlimited build minutes
- ✓ Unlimited repositories
- ✓ Unlimited collaborators

\$129

PER MONTH

Startup

BEST FOR SMALL TEAMS

2

 Concurrent jobs

- ✓ Unlimited build minutes
- ✓ Unlimited repositories
- ✓ Unlimited collaborators

Start Trial

\$249

PER MONTH

Small Business

GREAT FOR GROWING TEAMS

5

 Concurrent jobs

- ✓ Unlimited build minutes
- ✓ Unlimited repositories
- ✓ Unlimited collaborators

\$489

PER MONTH

Premium


PERFECT FOR LARGER TEAMS

10

 Concurrent jobs

- ✓ Unlimited build minutes
- ✓ Unlimited repositories
- ✓ Unlimited collaborators

ALL PRICES SHOWN IN USD



Jenkins to samodzielny program oparty na Javie, gotowy do uruchomienia natychmiast po zainstalowaniu, z pakietami dla Windows, Mac OS X i innych systemów operacyjnych typu Unix

Dzięki setkom wtyczek w Centrum aktualizacji Jenkins integruje się z praktycznie każdym narzędziem w ciągłej integracji i narzędziach do ciągłej dostawy.

Jenkins Pipeline. Jest to pakiet wtyczek, który obsługuje implementację i integrację ciągłych dostarczanych potoków do Jenkins. Pipeline zapewnia rozszerzalny zestaw narzędzi do modelowania prostych i złożonych potoków dostarczania „jako kod” za pośrednictwem DSL Pipeline.



Jenkins Plusy:

- Za darmo

- Dostosowywanie

- System wtyczek

- Pełna kontrola nad systemem

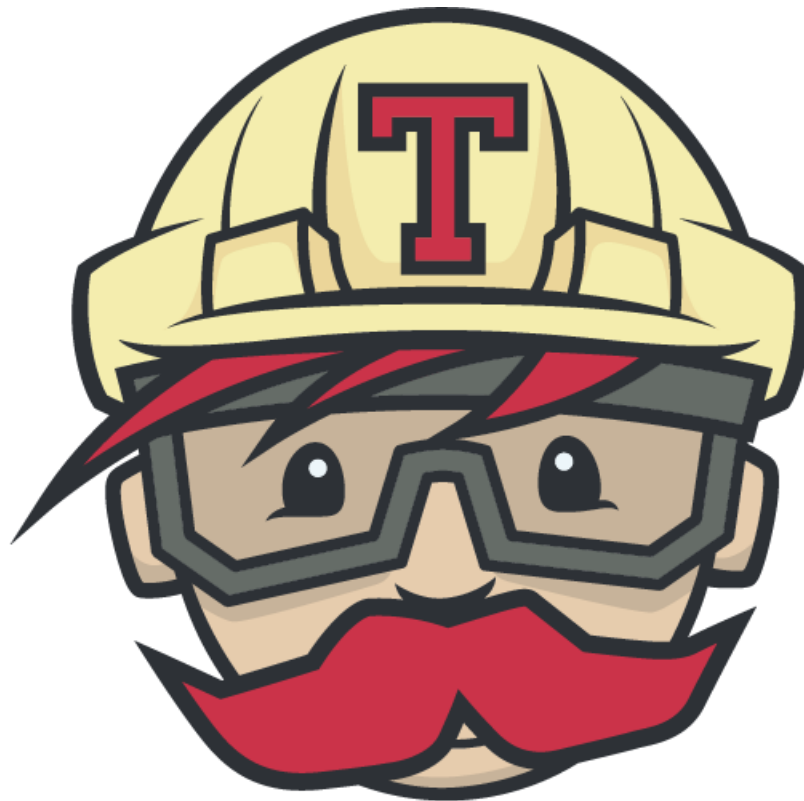
Wady Jenkins:


- Wymagany jest serwer dedykowany (lub kilka serwerów).

- Powoduje to dodatkowe wydatki.

- Czas potrzebny na konfigurację / dostosowanie

TRAVIS CI





Travis CI jest niemal tak samo szanowany jak Jenkins, i chociaż wiele jego komponentów jest open source, nie jest możliwe samodzielne hostowanie bez konta enterprise. Jednak korzystanie z Trávisa w projektach open source jest darmowe.

Każde polecenie, które chcemy uruchomić znajduje się w pliku `.travis.yml`, który znajduje się głównym repozytorium. Dzięki temu TravisCI pozwala na uruchamianie systemu CI z poziomu różnych branchy.

Travis jest dość prosty w obsłudze i działa bezpośrednio z repozytoriów znajdujących się na platformie GitHub.

Użyteczną cechą Trávisa jest możliwość jego uruchamiania na wielu systemach operacyjnych, co oznacza brak konieczności używania maszyn lub obrazów wirtualnych.



Travis CI Plusy:

- Szybki start

- Lekka konfiguracja YAML

- Bezpłatny plan dla projektów open source

Travis CI Minusy:

- Cena jest wyższa w porównaniu z CircleCI


Obsługiwane języki:

Android, C, C #, C ++, Clojure, Crystal, D, Dart, Erlang, Elixir, F #, Go, Groovy, Haskell, Haxe, Java, JavaScript (z Node.js), Julia, Objective-C, Perl, Perl6, PHP, Python, R, Ruby, Rust, Scala, Smalltalk, Visual Basic

GITLAB CI/CD



CI  CD



GitLab swoją działalność rozpoczął jako platforma służąca do hostingu kodu źródłowego dostępna w wersji open source.

W przeciwieństwie jednak do takich platform jak GitHub, GitLab zawiera teraz zaawansowaną implementację CI / CD (o nazwie AutoDevOps) wbudowaną w ich platformę.

Dla użytkowników, którzy używają platformy GitLab jako repozytorium kodu GitLab CI/CD jest dość dużą wygodą.

W celu korzystania z ww. systemu ciągłej integracji należy do repozytorium z kodem źródłowym dodać plik o nazwie `.gitlab-ci.yml`

GitLab CI/CD posiada również funkcję integracji z repozytoriami GitHub.

BAMBOO



Bamboo

CIĄGŁE DOSTARCZANIE

- Ciągłe dostarczanie (ang. continuous delivery) jest praktyką programistyczną, gdzie zespół wytwarza oprogramowanie w krótkich cyklach (np. dwutygodniowych sprintach), posiadając całą czas pewność, że każda kolejna zmiana w kodzie, bez względu na jej rozmiar, może zostać wydana (zaprezentowana użytkownikowi końcowemu) **w dowolnym momencie**. Głównym celem continuous delivery jest szybsze i częstsze budowanie, testowanie oraz wydawanie oprogramowania.
- *Ciągłe dostarczanie to praktyka programistyczna, gdzie oprogramowanie jest budowane w taki sposób, że może ono zostać opublikowane na środowisku produkcyjnym w dowolnej chwili.*

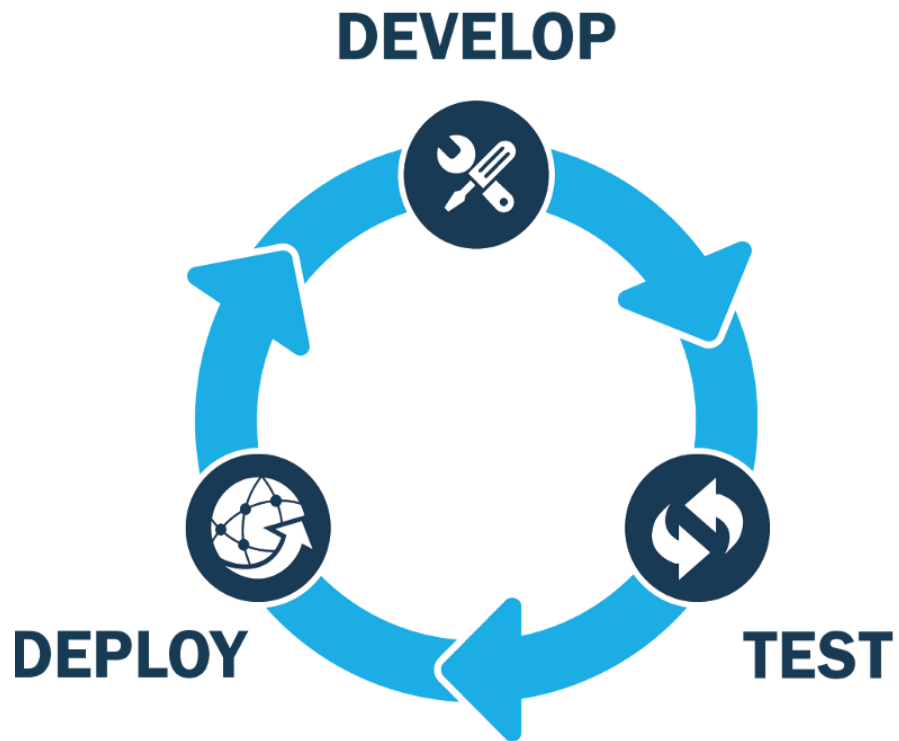


Bamboo to produkt CI/CD od Atlassian, firmy znanej w większości środowisk programistycznych z oprogramowania do śledzenia błędów w JIRA.

Jedną z kluczowych zalet Bamboo jest ścisła integracja z innymi produktami Atlassian (takimi jak JIRA i Bitbucket) dla tych, którzy już obsługują te systemy. Posiada również duży zasób plug-in'ów.

Z drugiej strony, Bamboo ma małą społeczność użytkowników, dzięki czemu użytkownicy mogą być bardziej zależni od wsparcia Atlassian.

DZIĘKUJĘ ZA UWAGĘ!



ŹRÓDŁA

- <https://www.exoscale.com/syslog/what-is-continuous-integration/>
- <https://productvision.pl/2016/continuous-integration/>
- https://pl.wikipedia.org/wiki/Ci%C4%85g%C5%82a_integracja
- <https://www.youtube.com/watch?v=ojr2Dy0Pjhw>
- <https://productvision.pl/2016/continuous-delivery/>