

## Program szkolenia:

# Continuous Integration

### Informacje:

|                        |                               |
|------------------------|-------------------------------|
| <b>Nazwa:</b>          | <b>Continuous Integration</b> |
| <b>Kod:</b>            | <b>tools-CI-CD</b>            |
| <b>Kategoria:</b>      | Narzędzia                     |
| <b>Grupa docelowa:</b> | developerzy<br>testerzy       |
| <b>Czas trwania:</b>   | 2 dni                         |
| <b>Forma:</b>          | 50% wykłady / 50% warsztaty   |

---

Podczas szkolenia uczestnicy poznają informacje oraz praktyczne techniki pozwalające na samodzielne projektowanie oraz implementowanie procesu ciągłej integracji.

W pierwszej kolejności zaprezentowane zostaną zagadnienia podstawowe takie jak: instalacja i konfiguracja narzędzi, automatyzacja kompilacji, różnych rodzajów testowania oraz statycznej analizy kodu.

Następnie poruszone zostaną zaawansowane rozwiązania, takie jak: integracja z systemami zewnętrznymi, współbieżne wykonywanie planów oraz wykorzystanie procesu ciągłej integracji podczas wdrażania koncepcji „Continuous Delivery”.

Na zakończenie uczestnicy poznają polecane metody projektowania i optymalizacji oraz wskazówki pozwalające wybrać narzędzia najbardziej odpowiadające ich potrzebom.

### Zalety szkolenia:

- Poprawienie jakości oprogramowania
- Automatyzacja zadań
- Continuous Delivery
- Zalecane wzorce i praktyki

## Szczegółowy program:

### 1. Podstawy

1.1. Co to jest ciągła integracja

1.2. Cechy

1.3. Korzyści

1.4. Ciągła integracja jako proces

1.4.1. Prezentacja praktyk programistycznych wymaganych do poprawnej implementacji CI, wykazanie, że CI to proces a nie tylko narzędzie do automatycznej kompilacji

### 2. Automatyzacja

2.1. Dlaczego warto automatyzować

2.1.1. Przedstawienie korzyści płynących z automatyzacji zadań

2.2. Narzędzia

2.2.1. Przedstawienie najpopularniejszych narzędzi umożliwiających automatyzację (Ant, Maven, Gradle i inne), ich zalet, wad oraz potencjalnych zastosowań.

2.3. Automatyzacja na serwerze

2.3.1. Opis możliwości automatyzacji oferowanej przez serwer CI

2.4. Prostota kontra elegancja

2.4.1. Przedstawienie dylematu pomiędzy stosowaniem rozwiązań szybkich a rozwiązań czasochłonnych i eleganckich, jakie są ich zalety, wady oraz kiedy warto je stosować

### 3. Serwer ciągłej integracji

3.1. Narzędzie

3.1.1. Pierwsze zaprezentowanie konkretnego serwera CI, np.: Atlassian Bamboo, przedstawienie procesu instalacji oraz podstawowych jego elementów i funkcji

3.2. Architektura

3.2.1. Przedstawienie architektury działającego systemu, jakie są jej cechy, jak przebiega komunikacja, jakie są możliwości jej modernizacji oraz jakie posiada wymagania środowiskowe

### 3.3. Podstawowe rozwiązania

3.3.1. W tej części przedstawiona zostanie implementacja najczęściej wykonywanych zadań. Są to: tworzenie planu automatycznie kompilującego kod źródłowy, jak pracować z wynikami planów, zarządzanie użytkownikami oraz zaprezentowanie dostępnych możliwości automatycznego powiadamiania i uruchamiania.

## 4. Testowanie

### 4.1. Wprowadzenie

4.1.1. Ogólne przedstawienie możliwości testowania za pomocą serwera CI

### 4.2. Korzyści

### 4.3. Testy jednostkowe

4.3.1. Przedstawienie możliwości wykonywania testów jednostkowych oraz sposobu pracy z wynikami

### 4.4. Testy funkcjonalne

4.4.1. Przedstawienie możliwości automatyzacji testowania funkcjonalnego za pomocą Selenium oraz RobotFramework'a, opis przebiegu i możliwości integracji z serwerem CI, przykłady użytkowania oraz pracy z wynikami

### 4.5. Testy wydajnościowe

4.5.1. Przedstawienie najpopularniejszych narzędzi umożliwiających przeprowadzanie testów wydajnościowych oraz jakie są możliwości ich integracji z serwerem CI

### 4.6. Użyteczne opcje

4.6.1. Opis użytecznych opcji wspomagających testowanie, oraz jak wygląda praca z wynikami testów (raportowanie błędów, opcje powiadamiania)

### 4.7. Optymalizacja

4.7.1. Przedstawienie praktyk i rozwiązań umożliwiających optymalizację wykonywania testów, aby wyniki były bardziej wiarygodne i szybciej dostępne

## 5. Statyczna analiza kodu

### 5.1. Narzędzia

5.1.1. Przegląd dostępnych narzędzi (PMD, Sonar, CheckStyle, FindBugs), jakie są ich zalety, wady oraz potencjalne zastosowania

### 5.2. Integracja

5.2.1. Opis procesu integracji z serwerem CI, przykłady użytkowania oraz pracy z wynikami

5.3. Korzyści

## 6. Zalecane praktyki

6.1. Przykłady optymalizacji

6.1.1. Przedstawienie przykładów oraz kluczowych praktyk i elementów które pozwolą uczestnikom na dalszą samodzielną optymalizację

6.2. Wzorce i anty-wzorce

6.2.1. Zaprezentowanie najczęściej występujących błędów projektowych oraz ogólnie sprawdzonych i rekomendowanych praktyk.

6.3. Repozytorium Maven'a

6.3.1. Przedstawienie możliwości wykorzystania repozytorium Maven'a w środowisku CI, jak również korzyści i wyzwań z tego płynących

6.4. Trudne problemy

6.4.1. Przedstawienie podstawowych i najczęściej występujących problemów oraz ich potencjalnych rozwiązania.

## 7. Zaawansowane praktyki

7.1. Rozproszone budowanie

7.1.1. Zaprezentowanie możliwości wykonywania planów w środowisku rozproszonym, jakie płyną z tego korzyści oraz jakie niesie to wyzwania

7.2. Integracja z systemami zewnętrznymi

7.2.1. Zaprezentowanie konkretnych przykładów użytkownika CI w środowisku Cloud czy też integracji z narzędziami zewnętrznymi, np.: Atlassian JIRA

7.3. Rozszerzanie funkcjonalności

7.3.1. Opis możliwości rozszerzania funkcjonalności popularnych serwerów CI za pomocą pluginów, tworzenie i konfiguracja

7.4. Współbieżność

7.4.1. Przedstawienie funkcjonalności współbieżnego wykonywania planów, przykłady zastosowań, korzyści i kosztów

7.5. Wspieranie repozytoriów

7.5.1. Omówienie wspieranych repozytoriów kodu źródłowego przez serwery CI, jakie oferują one funkcje pomocnicze, dodatkowo opis wsparcia dla DVCS(jak Git czy Mercurial), przykłady użytkowania

## 7.6. Automatyczna integracja baz danych

7.6.1. Przedstawienie koncepcji oraz korzyści z niej płynących, operacji bazodanowych podlegających automatyzacji oraz przykładowych rozwiązań /strategii

## 8. Continuous Delivery

### 8.1. Wprowadzenie

### 8.2. Korzyści

### 8.3. Co potrzebujemy

### 8.4. Zalecane praktyki

8.4.1. Przedstawienie zalecanych rozwiązań i pomysłów wspomagających stosowania Continuous Delivery

### 8.5. Automatyczne wdrażanie

8.5.1. Przedstawienie podstawowych informacji oraz korzyści, podanie przykładów konkretnych implementacji dla serwerów Tomcat, WebSphere, WebLogic

## 9. Przegląd dostępnych rozwiązań

### 9.1. Przegląd

9.1.1. Zaprezentowanie najciekawszych obecnie dostępnych serwerów CI( Atlassian Bamboo, Hudson/Jenkins, AntHillPro i inne) oraz elementów które je wyróżniają

### 9.2. Samodzielny wybór

9.2.1. Przedstawienie najważniejszych cech i rad które pozwalają na wybranie odpowiedniego narzędzia przez uczestnika kursu

### 9.3. Build pipe lines tools

9.3.1. Przedstawienie alternatywy dla skomplikowanych serwerów w postaci prostych rozwiązań jak np.: BuildBot, omówienie zalet, wad i potencjalnych zastosowań