



Manuel Pichler  
<http://www.manuel-pichler.de>

PHP Unconference  
26. April 2008



- Manuel Pichler
  - Jahrgang 1978
  - Diplom Informatiker
  - Entwickler bei der i-world GmbH
  - Autor von PHP\_Depend und der CruiseControl-Erweiterung phpUnderControl
  - Beteiligt bei dem einen oder anderen Open Source Projekt: torii, phidget

# Was ist Continuous Integration?



- Continuous Integration stellt den fortlaufenden Abgleich zwischen den Entwicklungsständen in einem Projekt sicher.
- Das Einchecken von Änderungen/Neuerungen in die Versionsverwaltung führt zu einem automatischen Integrationsbuild.
- Neben der Ausführung automatisierter Tests können mit jedem Build verschiedene Analysen automatisiert durchgeführt werden.

# Was bringt Continuous Integration?



- Bei getesteter Software können entstandene Inkompatibilitäten zwischen Komponenten frühestmöglich entdeckt werden.
  - Durch schnelles Feedback an die Entwickler können Fehler unverzüglich behoben werden.
  - Entwickler arbeiten sehr nah am aktuellen Stand des Projekts mit ähnlichen Arbeitskopien.
- Der Buildprozess muss automatisiert werden. Dies reduziert den Aufwand für Entwickler und hilft Fehler zu vermeiden.

# Was bringt Continuous Integration?



- Reduzierter Aufwand für die Integration einzelner Komponenten.
- Durch die Einbindung eines Werkzeugs wie phpdoc, existiert zu jedem Zeitpunkt eine aktuelle API-Dokumentation.
- Automatische Prüfungen des Quelltext auf Einhaltung der Coding-Standards stellt eine gleichbleibende Qualität sicher.

# Was bringt Continuous Integration?



- Generierte Metriken geben unmittelbar Auskunft über mögliche Schwachstellen der Software.
  - Metriken decken viele Schwachstellen auf, sie sollten aber nicht zum heiligen Gral werden.
- Automatisierte Builds ermöglichen es eine Applikation regelmäßig auf verschiedenen Plattformen und Umgebungen zu testen.
- Es entstehen regelmäßig (meist mehrmals täglich) lauffähige Versionen des Projekts.



# CI - Best Practices (1)

- Entwickler sollten Änderungen in regelmäßigen Abständen einchecken (mehrmals täglich)
  - Fehler und Konflikte zwischen den Änderungen einzelner Personen werden schnell sichtbar, bevor es zu einem Problem wird sie zu beheben.
- Ein Commit gilt erst dann als erfolgreich, wenn der Build auf dem CI-Server abgeschlossen ist.
  - Die Entwickler, die einen Build auslösten, sind für die Behebung möglicher Fehler verantwortlich.
  - Also kein <Enter> und „Feierabend“



# CI - Best Practices (2)

- Feedback über Erfolg oder Misserfolg sollte möglichst schnell gegeben werden.
  - Eine ausreichende Hardwareausstattung des CI-Servers ist erforderlich.
  - Die Entwicklung sollte in kleinen Schritten erfolgen und regelmäßig committet werden.
  - Zeitaufwändige Bestandteile des Buildprozesses gehören in einen nachgestellten, zweiten Build.
    - Tests gegen eine Datenbank
    - Analysen(CodeSniffer, Checkstyle) des Quelltextes



# CI - Best Practices (3)

- In einem zweiten Schritt wird ein vollständiger Build des Projektes durchgeführt.
  - Identische Build- u. Zielplattform
  - Zeitaufwändiger Build der letzten stabilen Version
  - Fehler führen zu einer Anpassung der Tests
- Einheitliche Infrastruktur aller Beteiligten
  - Verwendete Bibliotheken und Datenbankschemata gehören in die Versionsverwaltung oder sollten bspw. mittels `svn:externals` eingebunden werden.



# CI - Best Practices (4)

- Feedback sollte nicht in Spam ausarten.
  - Nachrichten sollten nur an beteiligte Personen versandt werden.
- Gimmicks visualisieren den Buildstatus.
  - Ant Ambient Orb Task
  - CC X10 Publisher



Quelle: [http://www.ambientdevices.com/cat/images/GreenOrb\\_onwhite.jpg](http://www.ambientdevices.com/cat/images/GreenOrb_onwhite.jpg)



# Welche Werkzeuge gibt es?

- Continuous Intregation Server
  - CruiseControl
    - phpUnderControl
  - Hudson
  - Xinc
  - Continuum
  - Bamboo
  - ...



- Eine auf PHP abgestimmte Erweiterung des Continuous Integration Werkzeugs CruiseControl.
  - ursprünglich ein privates Patchset für CruiseControl.
  - Hauptziele ist ein leichter Einstieg in die Thematik.
  - Zur Zeit unterstützte PHP Entwicklungstools:
    - PHPUnit
    - phpDocumentor
    - PHP\_CodeSniffer



- Der Fokus für kommende phpUnderControl Versionen liegt auf:
  - Master/Slave-Projekte: Ein Projekt aggregiert die Test-Ergebnisse und Metriken verschiedener Projekte.
    - Ermöglicht das Testen mit verschiedenen PHP-Versionen und unterschiedlichen Betriebssystemen.
    - Erlaubt realistischere Coverage-Reports für Versions und Betriebssystem spezifischen Code.
  - Support für PHP\_Depend.
    - Analysiert Abhängigkeiten zwischen Softwarebestandteilen
    - Aktuell befinden sich CodeRank-Metriken in der Entwicklung.
    - Die Übernahme vieler Metriken aus PHPUnit ist geplant.



# Links

- **Continuous Integration**  
<http://martinfowler.com/articles/continuousIntegration.html>
- **Continuous Integration anti-patterns**  
<http://www.ibm.com/developerworks/java/library/j-ap11297/>  
<http://www.ibm.com/developerworks/java/library/j-ap03048/>
- **Ant Ambient Orb Task**  
<http://qualitylabs.org/projects/ambientorb/>
- **CI Feature Matrix**  
<http://confluence.public.thoughtworks.org/display/CC/CI+Feature+Matrix>