TeamCity dokumentacja

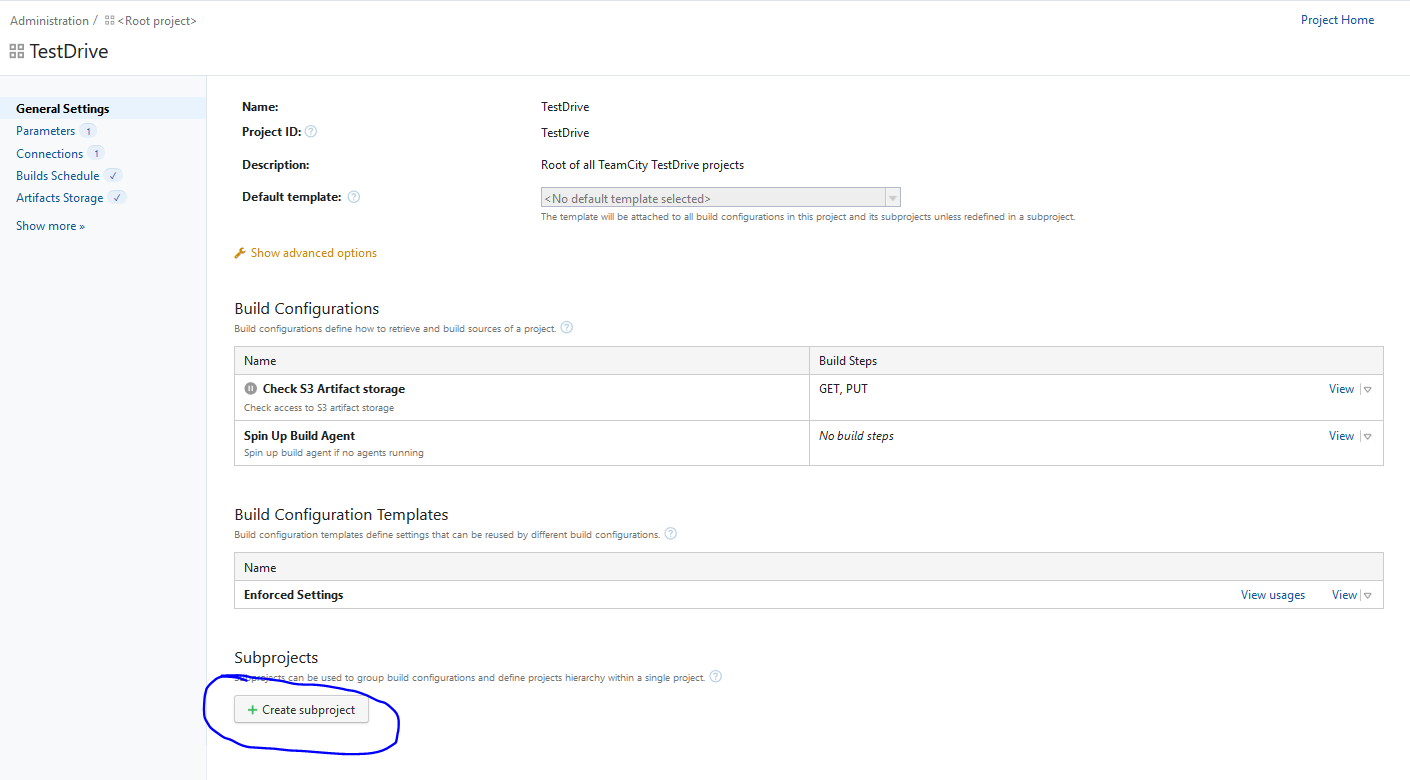
# Hosting serwera

Jako narzędzie CI do naszego projektu wykonywanego na przedmiocie ZPI wybraliśmy TeamCity, stworzone przez firmę jetbrains. Firma ta umożliwia wypróbowanie swojego narzędzia przez okres 60 dni. Skorzystaliśmy więc z tej okazji, i uruchomiliśmy swoją instancję TeamCity na ich serwerach w chmurze. Aby skorzystać z ich oferty, należy się zarejestrować i wejść w [www.jetbrains.com/teamcity/download/#section=testdrive](http://www.jetbrains.com/teamcity/download/#section=testdrive). Po naciśnięciu „Test driver in Cloud” zostaniemy przekierowani na stronę instruującą nas co zrobić dalej.

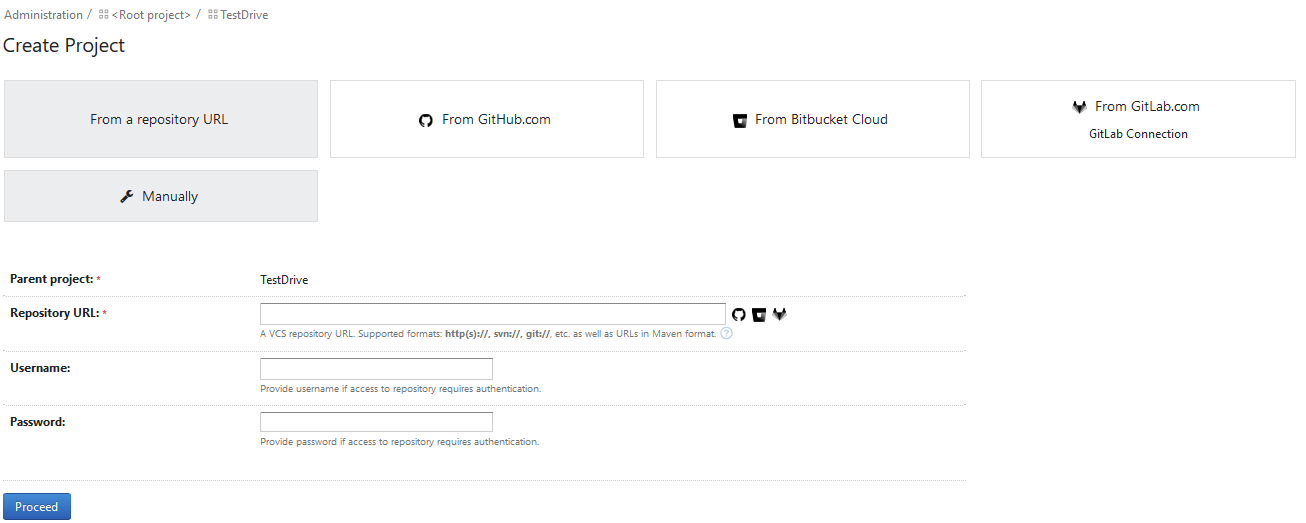
# 2. Stworzenie projektu

Wybranie darmowej opcji hostingu w chmurze wiążę się z jednym, tutaj pojawiającym, niedogonieniem. Podczas tworzenia projektu musimy go stworzyć jako subprojekt projektu o nazwie „TestDrive”. Nie wprowadza to jednak żadnych różnic w dalszej konfiguracji.

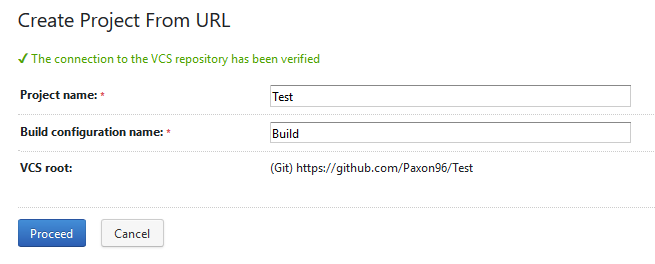
Aby utworzyć wspomniany subprojekt, wchodzimy w ‘Administration’ znajdujące się w prawym górnym roku, tuż obok naszej nazwy użytkownika i wybieramy projekt o nazwie „TestDrive”. Przycisk umożliwiający nam stworzenie subprojektu zaznaczony jest poniżej.



Po wybraniu w/w opcji zostajemy przekierowani do okna, w którym łączymy projekt z repozytorium.



Po poprawnym wypełnieniu powyższych pól zostajemy poinformowani o poprawnie stworzonym połączeniu do repozytorium. W tym miejscu możemy nadać nazwę naszemu projektowi i konfiguracji budowania.



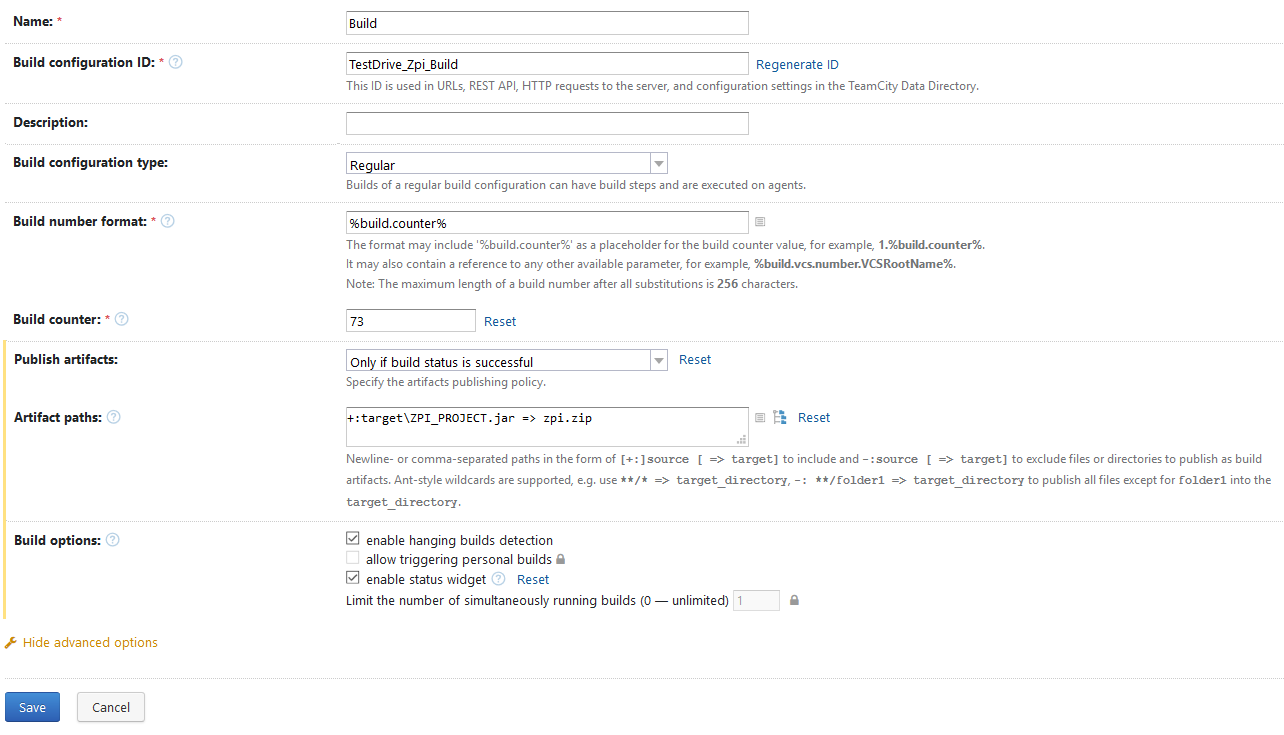
# Konfiguracja budowania projektu

Po stworzeniu projektu zostaniemy automatycznie przeniesieni na stronę konfiguracji budowania.

Jeśli niżej wymienione opcje są ukryte, należy nacisnąć „[Show advanced options](javascript://)” znajdujące się pod koniec strony.

## General Settings

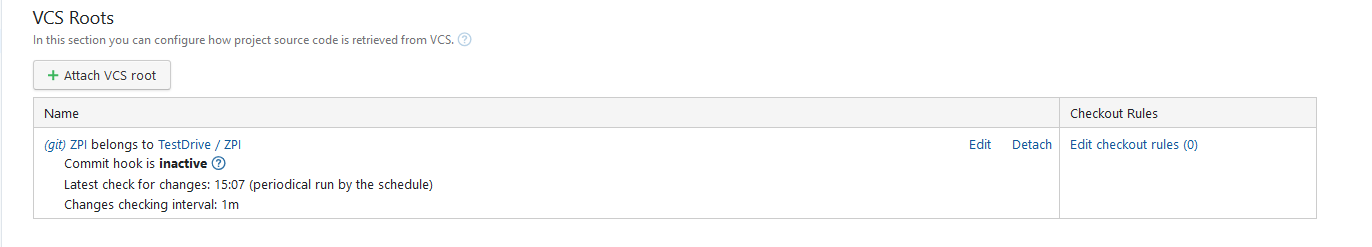
W tej zakładce konfigurujemy podstawowe ustawienia naszej konfiguracji budowania projektu. Możemy zmienić jej nazwę, wygenerować nowe ID konfiguracji. Ale najważniejszą rzeczą tutaj są ustawienia publikowania artefaktów. Bazując na naszym projekcie, ustawienia wyglądają następująco:



Publikacja artefaktów jest ustawiona tak, że zostają one publikowane tylko gdy build zakończy się powodzeniem. W polu „Artifact paths” wskazujemy agentowi budującemu, które pliki z całego projektu ma załączyć do publikowanych artefaktów, w naszym przypadku jest to tylko wykonywalny JAR, opakowywany w zip’a.

## Version Control Settings

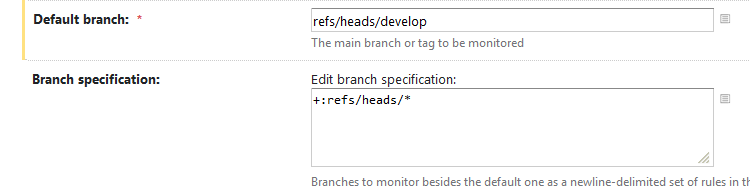
W zakładce „[Version Control Settings](https://teamcity.jetbrains.com/admin/editBuildTypeVcsRoots.html?init=1&id=buildType:TestDrive_Zpi_Build)” konfigurujemy domyślne ustawienia połączenia z repozytorium, tak by dostosować TeamCity do GitFlow. Wybieramy stworzone już ustawienia VCS.



Na potrzeby naszego projektu musieliśmy zmienić domyślne ustawienia w kilku miejscach, jako że tych ustawień jest naprawdę wiele, uwzględnię tylko te które zostały zmienione.

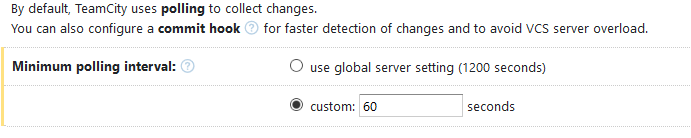
### Ustawienia branchy

Musimy wskazać jako główną gałąź developa, oraz w „branch specification” ustawić TeamCity tak, żeby był budowane tylko head’y gałęzi. W przeciwnym wypadku budowany będzie każdy commit, co znacząco spowalnia weryfikację budowania po stronie repozytorium.



### Czas odpytywana repozytorium

TeamCity domyślnie odpytuje repozytorium co 1200 sekund, sprawdzając czy nie zaszły jakieś zmiany. W celu usprawnienia tego procesu można zmienić tą wartość na 60 sekund.



## Kroki budowania

TeamCity umożliwia autodetekcję kroków budowania na podstawie plików znajdujących się w repozytorium. Jest np. wstanie wykryć plik POM.xml zawierający konfigurację marena.

Aby przeszukać repozytorium pod kątem kroków budowania wystarczy nacisnąć:

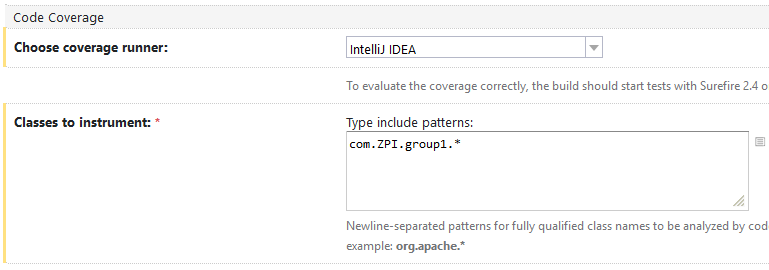


Po wybraniu sugerowanej konfiguracji, należy wprowadzić w niej kilka zmian.

Należy zmienić polecenie mavena na to załączone poniżej, aby plugin odpowiedzialny za zbudowanie wykonywalnego JAR’a się uruchomił.



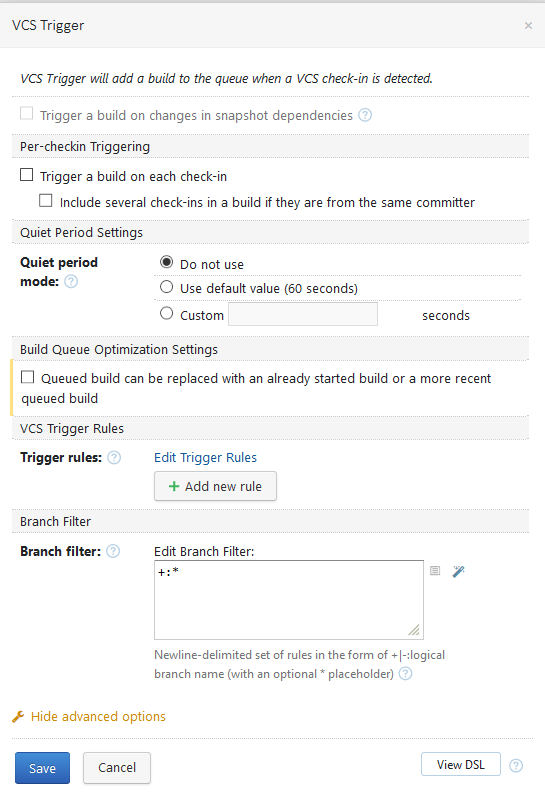
Dodatkowo, dla lepszej kontroli nad procesem powstania projektu dobrze jest ustawić analizę kodu pod względem pokrycia testami. Wystarczy domyślną konfiguracją zmienić na tą niżej pokazaną.



## Wyzwalacz na wprowadzone zmiany

Zakładce „Triggers” musimy dodać nowy wyzwalacz, tak aby TeamCity reagowało na zmiany wprowadzone w każdej gałęzi.

Naciskamy „Add new trigger” i wybieramy z rozwijanej listy „VCS Trigger”. Konfigurujemy go tak, jak to jest pokazane poniżej.



## Build Features

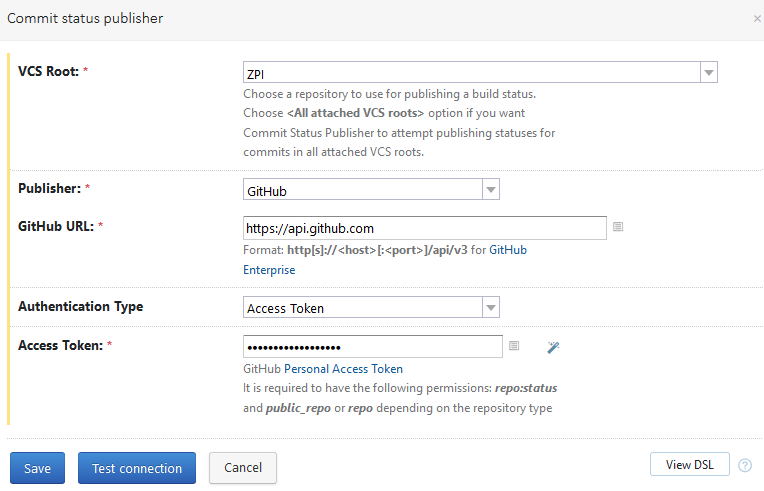
Dodatkowe właściwości budowania pomogą nam w lepszym zarządzaniu CI w projekcie. Jedno jest ustawione domyślnie i zarządza czyszczeniem plików przed przystąpieniem do budowania projektu. Należy dodać jeszcze trzy.

### 

### Commit status publisher

Umożliwia TeamCity wysłanie powiadomienia do repozytorium o statusie budowanie, tzn. czy budowanie jest w trakcie, zakończyło się ze skutkiem negatywnym lub pozytywnym.

Po naciśnięciu „Add build feature” i wybraniu commit status publisher konfigurujemy go tak jak poniżej.

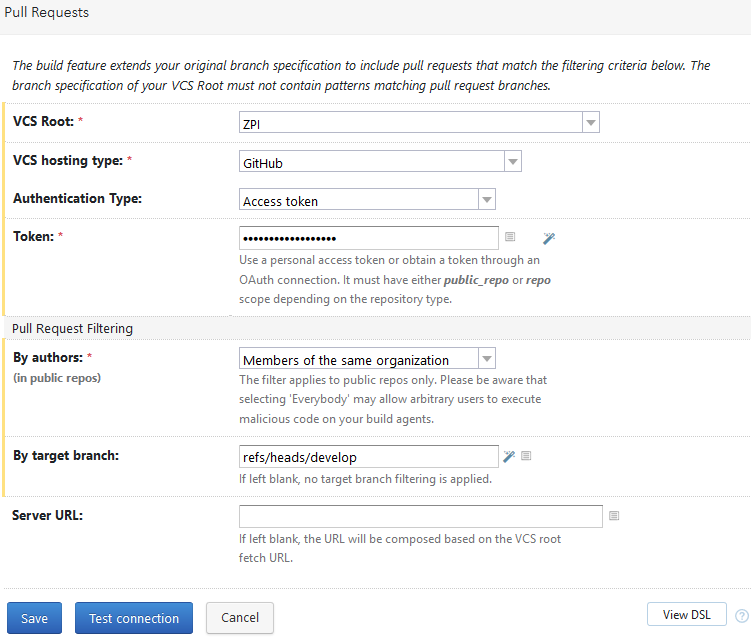


VSC Root wybieramy z rozwijanej listy, jest to połączenie do naszego repozytorium. Następne opcje są zależne od wybranego Publishera, w tym przypadku tak wyglądają opcję po wybraniu GitHub.

### Pull Request

Dzięki tej opcji, wymuszamy w repozytorium aby pull request skierowany do wskazanej gałęzi był poprawnie zbudowany.

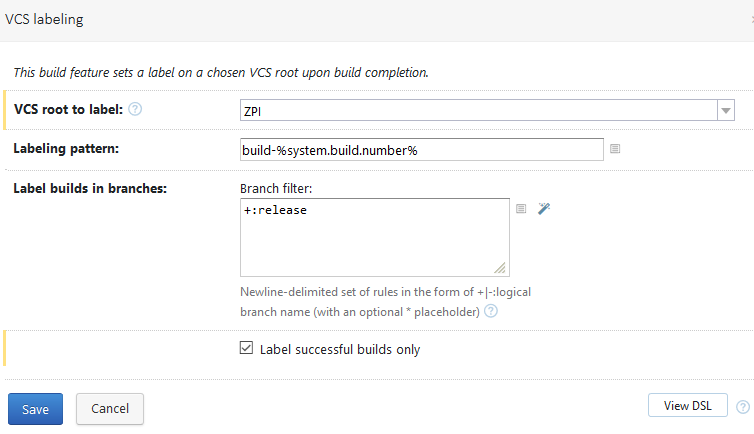
Konfiguracja:



### VCS Labelling

Umożliwia stworzenie taga na repozytorium po poprawnym zbudowaniu projektu na wskazanej gałęzi. W naszym przypadku, na release.

Konfiguracja:



Tekst w polu Labeling pattern odpowiada nazwie stworzonego Tagu. W naszym przypadku jest to ‘build-<numer\_builda>’