



UNIwersytet Gdański

# Wprowadzenie do obiektów imitacji - mocki

---

Mateusz Miotk

09 kwiecień 2018

Instytut Informatyki UG

# Zadanie obiektów imitacji

Zadaniem testów jednostkowych jest sprawdzenie działania pojedynczej metody. A co w przypadku kiedy działanie tej metody jest uzależnione od czynników zewnętrznych np.: funkcjonowanie bazy danych czy silnika serwletów?

Wówczas testy muszą inicjować prawie wszystkie komponenty systemu zewnętrznego, a jest to wyjątkowo pracochłonne i ponadto wprowadzi do testów zbyt wysoki stopień powiązania z systemem: wystarczy, że zmiana np.: tabeli w bazie danych spowoduje, że przetestowany kod przestanie działać.

Aby uniknąć tych zagrożeń wymyślono tzw. **obiekty imitacji**.

# Definicja obiektów imitacji

**Obiekt imitacji** (lub inaczej atrapa) to obiekt symulujący określone zachowania, którym można zastąpić rzeczywiste obiekty. Atrapy są bardzo przydatne, gdy obiekty zależne od zewnętrznych zasobów nie mogą uzyskać dostępu do tych zasobów.

Ponadto są przydatne w wielu innych sytuacjach m.in:




- Obiekt rzeczywisty zachowuje się niedeterministycznie
- Obiekt rzeczywisty jest trudny do skonfigurowania
- Trudne jest wywołanie odpowiedniego zachowania obiektu
- Działanie rzeczywistego obiektu jest powolne
- Test musi uzyskać informacje o sposobie użycia rzeczywistego obiektu
- Obiekt rzeczywisty jeszcze nie istnieje

Są znane ogólnie trzy metody tworzenia obiektów imitacji:

- Ręcznie
- Przy pomocy narzędzia Mockito
- Przy pomocy narzędzia EasyMock

## Porady przy stosowaniu obiektów imitacji

- **Nigdy nie testujemy imitacji.** One mają nam umożliwić testowanie innych składników aplikacji w czasie normalnego działania programu. Testowanie ich jest pozbawione sensu, bo poddajemy testom kod, który nie będzie uruchamiany w czasie pracy.
- **Działanie klasy imitacji musi być identyczne z działaniem imitowanej klasy.** W przeciwnym przypadku klasa imitacji nie będzie odzwierciedlały prawdziwego jego działania.

-  V. Farcic, A. Garcia, *TDD. Programowanie w Javie. Sterowanie testami*, Wydawnictwo Helion, 2016.
-  A. Hund D. Thomas, *JUnit. Pragmatyczne testy jednostkowe w Javie*, Wydawnictwo Helion, 2006.
-  R. Sokół, *Testowanie aplikacji Java za pomocą JUnit*, Wydawnictwo Helion, 2017.