Podstawy inżynierii oprogramowania

dr inż. Krystian Wojtkiewicz

Software Quality Assurance
Zapewnianie Jakości Oprogramowania

Podstawowe pojęcia

Możemy poczuć lub ocenić, ale nie zmierzyć?

„**I know it when I see it**” – implikuje jakoby jakości nie dało się kontrolować, zarządzać nią czy zmierzyć

Często ocena jest pod wpływem odczuć, a nie faktów

Podstawowe pojęcia

W zawodzie Software Developera istnieje etyczny imperatyw jakości

Jakość to nie jest jedynie kwestia marketingu i postrzegania.

To jest zobowiązanie etyczne i prawne – ciąży na nas odpowiedzialność zawodowa związana z wytwarzanym oprogramowaniem

Profesjonaliści musza być w stanie wykazać i mieć pewność, że stosują „najlepsze praktyki”

Podstawowe pojęcia

W praktyce musimy umieć mierzyć jakość produktu.

O jakości możemy mówić w kontekście:

- **Zgodności z wymaganiami** (terminy, koszt)
- **Przydatność** – czy produkt spełnia założone zadanie?
- **Bezawaryjność**, niezawodność, solidność
- **Satysfakcja klienta** – czy użytkownicy są zadowoleni?

Czy jakość oprogramowania to…

Stopień w jakim system, komponent lub proces odpowiada wyspecyfikowanym wymaganiom.

Stopień w jakim system, komponent lub proces odpowiada klientowi, jego potrzebom lub oczekiwaniom.

Wymiary jakości oprogramowania

Parametry techniczne	Poprawność	Brak błędów lub defektów	Mierzone w kontekście liczby błędów
	Niezawodność	Nie zawiesza się zbyt często	Mierzone w kontekście liczby zawieszeń (failure, crash)
	Wydajność	jest szybkie i małe	Mierzone w kontekście szybkości i zużycia miejsca
	Łatwość konserwacji	Łatwe do zaadaptowania do nowych wymagań	Mierzone w kontekście logów zmian i analizy wpływu (lines affected, time ans efford)
	„zdolność”	Robi to co ma robić	Mierzone w kontekście pokrycia wymagań
Parametry zorientowane na użytkownika	Użyteczność	Jest odpowiednio wygodne dla docelowego użytkownika	Mierzone w kontekście satysfakcji użytkownika
	Łatwość instalacji / wdrożenia	Jest wygodnie i szybko można je zainstalować	Mierzone w kontekście liczby problemów zgłoszonych w procesie instalacji
	Dokumentacja	Jest dobrze udokumentowane	Mierzone w kontekście satysfakcji użytkownika
	Dostępność	Jest łatwo dostępne lub dostępne gdy potrzebne	Mierzone w kontekście satysfakcji użytkownika

Uzyskiwanie odpowiedniej jakości

Jakość produktu czy oprogramowania nie zdarza się przez przypadek

Aby uzyskać odpowiednią jakość musimy planować to od początku i monitorować ją ustawicznie (każdego dnia)

Wysoka jakość wymaga dyscypliny

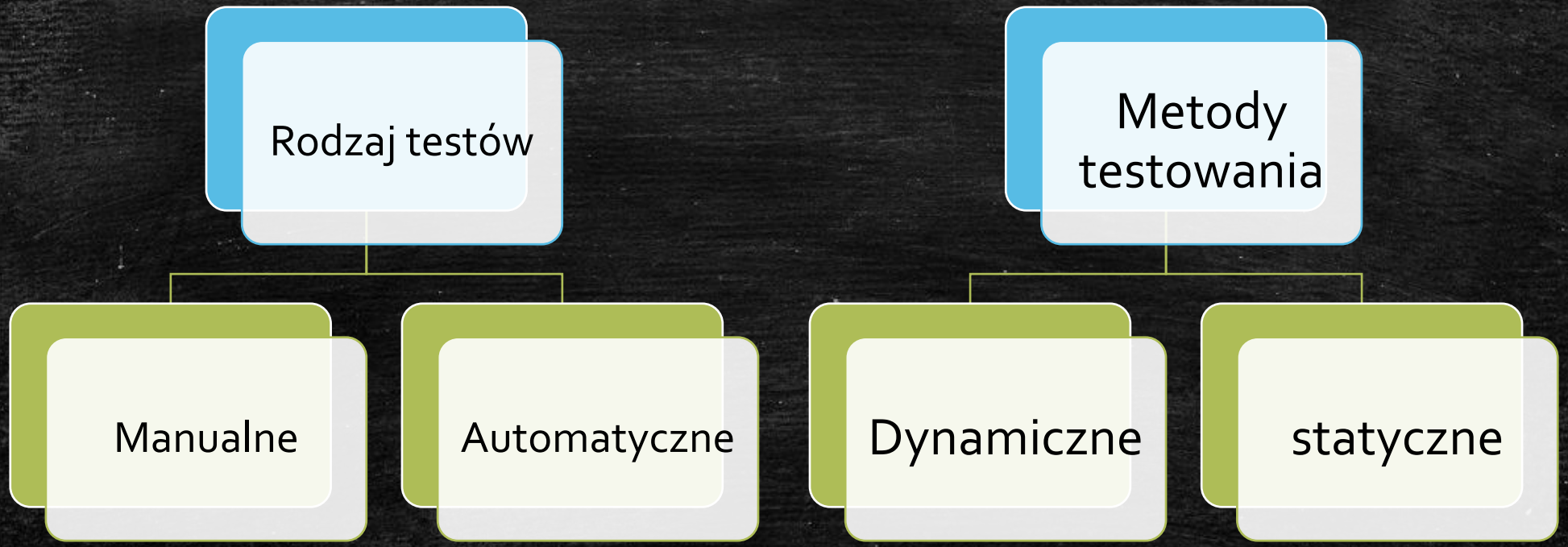
Metody uzyskiwania wysokiej jakości wyników są przedmiotem tzn. Quality Assurance (QA)

Główne zasady QA

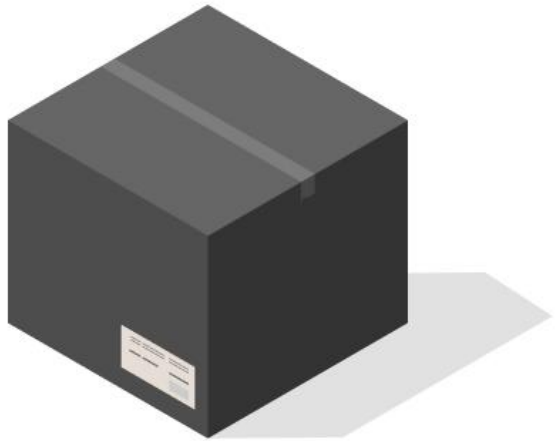
Musisz wiedzieć **co robisz**

Musisz wiedzieć **co powinieneś robić**

Musisz wiedzieć jak **zmierzyć różnicę**


Rodzaj testów
Manualne
Automatyczne
Metody testowania
Dynamiczne
statyczne

QA testers



Black box - we do not know anything

Developers

White box - we know everything


Black box Testing
White box Testing
Gray box Testing
+
=
Testing Techniques To Stay Bug-Free

Podejścia do testowania

White box

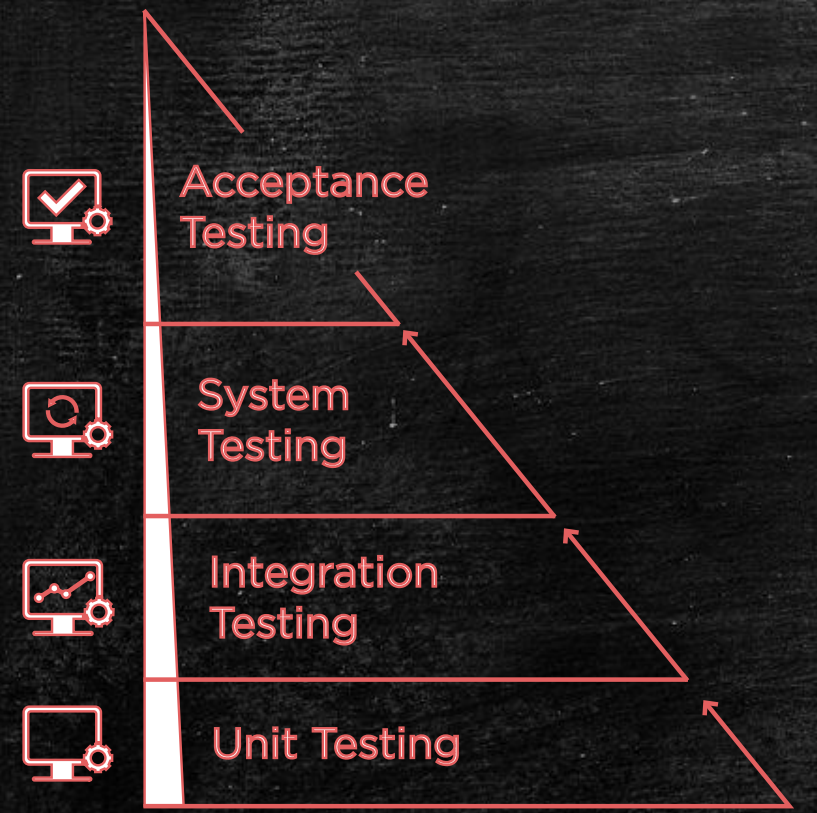
- Dokładność wynikająca z bezpośredniego przełożenia na strukturę kodu źródłowego.
- Łatwość optymalizacji owego kodu, dzięki jasnej identyfikacji wąskich gardeł.
- Łatwość zautomatyzowania.
- Inżynieryjna klarowność procedury: jej postępu i zakończenia

Black Box

- Mają większą szansę wykrycia błędów, ale jednocześnie nie dostarczają precyzyjnej informacji na temat przyczyny wystąpienia błędu w programie.
- Mają wykrywać błędy implementacji funkcjonalności zawartych w specyfikacji wymagań
- Przypadki testowe oparte są na specyfikacjach i wymaganiach, tzn. na tym co program ma robić. Zazwyczaj pochodzą one z zewnętrznych opisów oprogramowania, specyfikacji wymagań lub ustaleń projektowych.

Grey Box

- Jest to kombinacja testów biało i czarnoskrzynkowych



Levels of Software Testing

LeewayHertz

Testy jednostkowe

Wykonywane przez programistów na najniższym poziomie kodu, testy jednostkowe sprawdzają poprawność pojedynczych modułów lub funkcji oprogramowania. Celem jest identyfikacja błędów na wczesnym etapie.

Testy integracyjne

Ten poziom sprawdza interakcje między modułami lub komponentami oprogramowania, które zostały wcześniej przetestowane na poziomie jednostkowym. Celem jest wykrywanie błędów integracji między poszczególnymi elementami systemu.

Testy systemowe

Wykonuje się na już zintegrowanym systemie. Celem jest sprawdzenie, czy oprogramowanie spełnia wymagania funkcjonalne i niefunkcjonalne oraz czy działa poprawnie jako całość. Testy systemowe skupiają się na zachowaniu oprogramowania w różnych warunkach.

Testy akceptacyjne

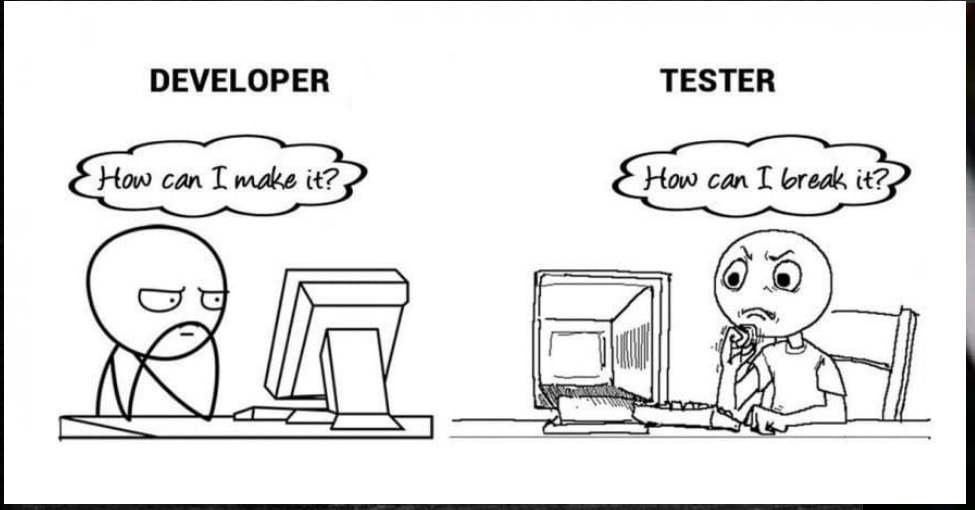
Przeprowadzane są przez klienta lub użytkownika końcowego, aby ocenić, czy oprogramowanie spełnia ich oczekiwania i wymagania. Celem jest potwierdzenie gotowości oprogramowania do użycia.

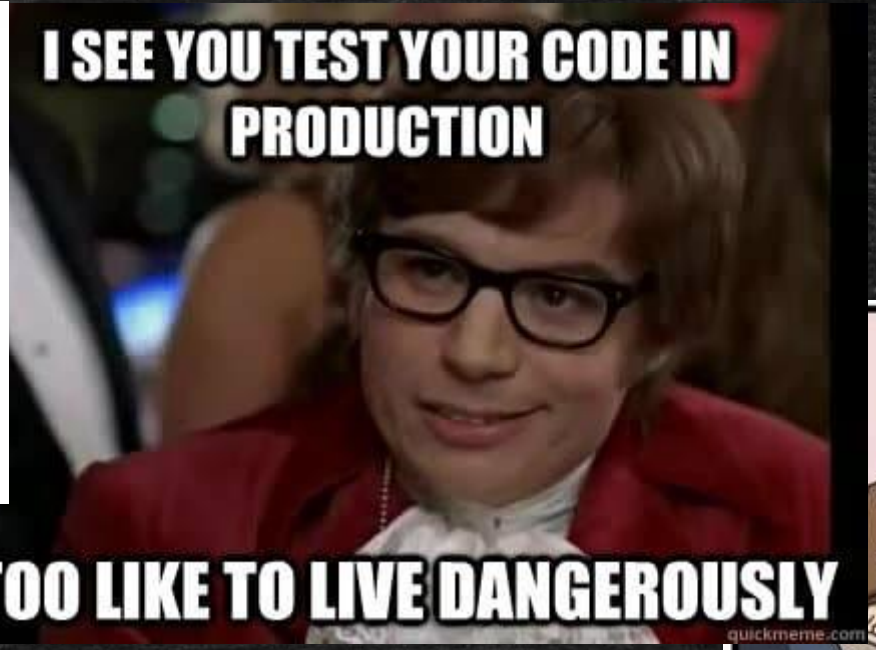
Testy wydajnościowe

Koncentrują się na ocenie wydajności i skalowalności oprogramowania w różnych warunkach obciążeniowych. Testy te pomagają zidentyfikować potencjalne problemy wydajnościowe i zoptymalizować działanie systemu.

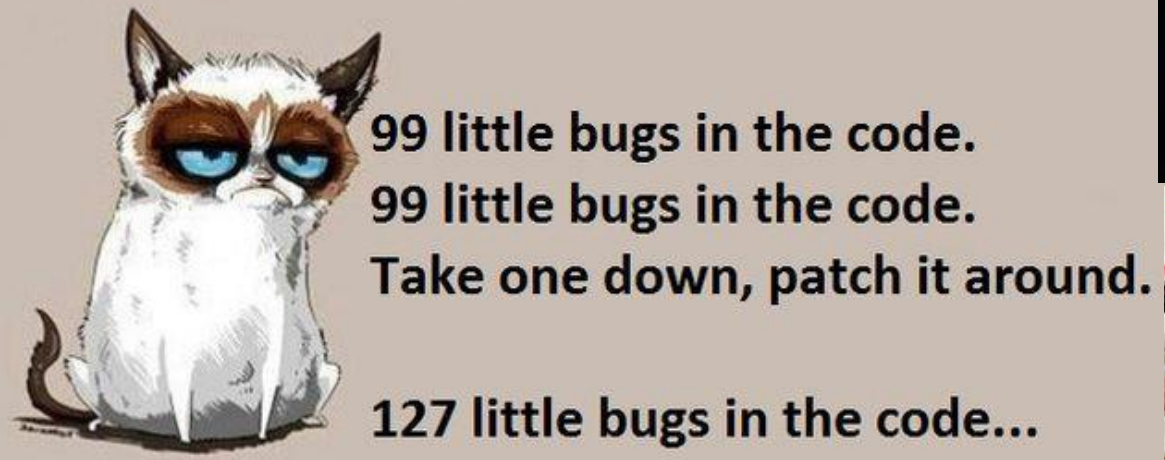
Testy zabezpieczeń

Mają na celu sprawdzenie odporności oprogramowania na ataki i zagrożenia. Testy te identyfikują luki w zabezpieczeniach i pomagają w wzmocnieniu ochrony systemu.


DEVELOPER
TESTER
How can I make it?
How can I break it?


I SEE YOU TEST YOUR CODE IN PRODUCTION
I TOO LIKE TO LIVE DANGEROUSLY
quickmeme.com


How can we improve our app


99 little bugs in the code.
99 little bugs in the code.
Take one down, patch it around.
127 little bugs in the code...


They told me to
SCAN this laptop to
remove virus


Add new features
Change the design
Fix all open bugs


PLEASE DON'T DO THAT
IT'S CALLED THE LIGHTSABER CHALLENGE. ALL THE KIDS ARE DOING IT
imgflip.com


imgflip.com