1 Fakes

Fakes er noget vi anvender for at isolere vores UUT. Vi opsætter miljøet omkring vores test. Fx hvis vores test anvender et resultat fra en anden funktion i en anden klasse, så har vi brug for at kontrollere, hvad resultatet bliver for at kunne kontrollere testen. Eller hvis vores funktion, som vi tester skal kalde en anden funktion, så skal vi sikres os at den kan kalde den rigtige. Derfor anvender vi 2 typer af fakes, stubs og mocks.
Når vi arbejder med fakes sætter det krav til vores design af systemet. Vi skal anvende interfaces i vores design, da vi på den måde kan styre hvilken klasse, som opfylder interfaces skal vi anvende. Vi skal tænke på seperation of concerns og lav kobling i vores design, når vi skal anvende fakes.

2 State based test - STUBS

Stubs er en en state based test. Vi tester altså på, hvilket state vores UUT er i. Når vi arbejder med stubs, arbejder vi med de 3 A'er:

- Arrange - her opsætter vi stubs til at returner det vi ønsker.
- Act - får vores UUT til at gøre det vi ønsker
- Assert - asserter på om vi får vores forventet resultat.

Vi asserter altså på om vores UUT er i det rigtige state og ikke på vores stubbe, da stubbe aldrig kan få tests til at fejle.

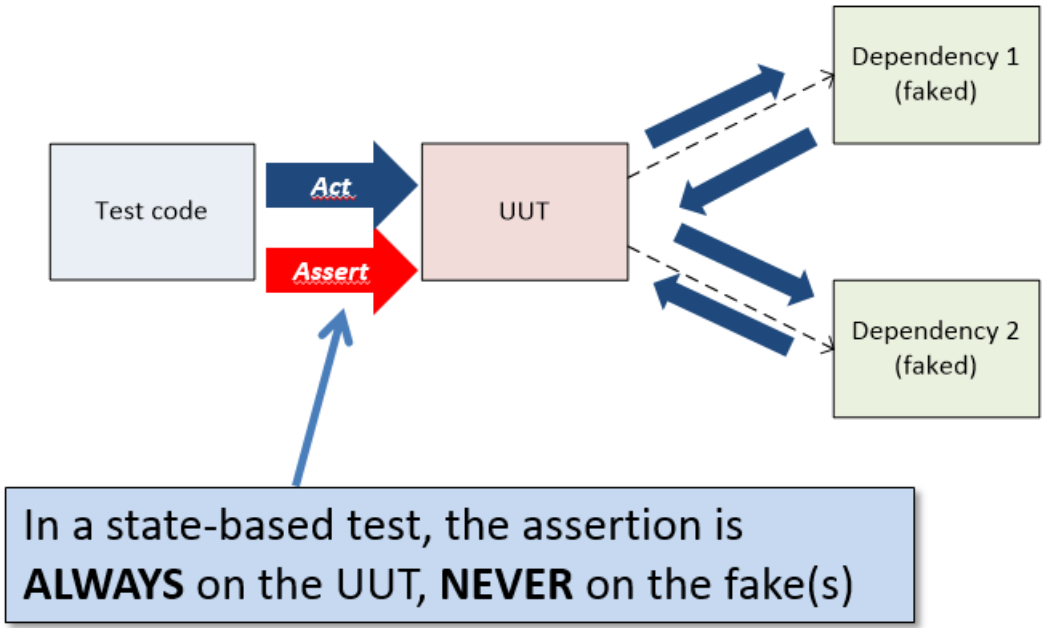

Figure 1:

2.0.1 Contructor injection

Eftersom STUBS handler om at teste stadier. Derfor kan det være en fordel at anvende constructor injection for du da vil have større kontrol over den fake som arbejder med din UUT hvis du har en medlamsvariabel med din fake, i din UUT.

2.0.2 Configurable stubs

En configurable stub er hvor du ved hjælp af set metoder kan bestemme return værdien på en anden metode i den fake.

3 Interaction based test - MOCKS

Mocks er interaction based test, hvilket betyder at vi ikke tester på hvilket state vores UUT er i, som i stubs, men på hvordan den opfører sig. Behavioral testing.

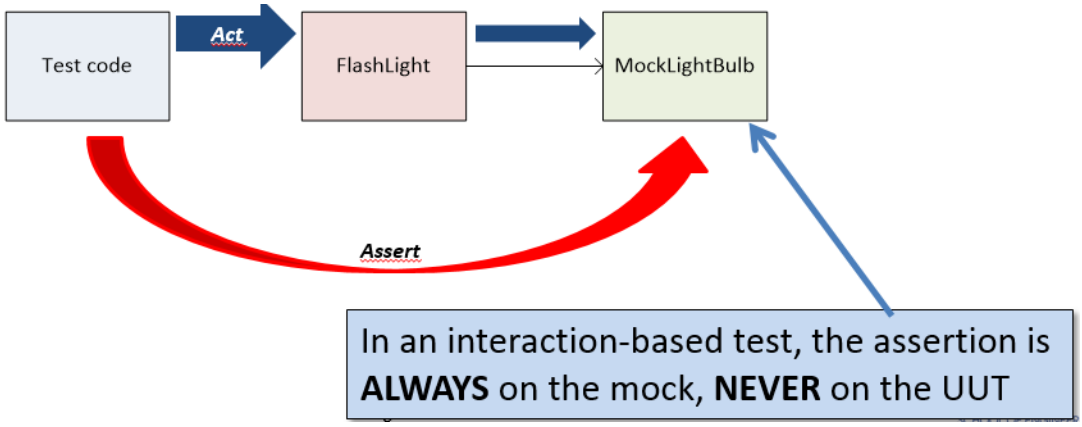

Figure 2: Hvordan man act'er og assert'er på mocks.

Som figuren illustrerer assert'er vi på mock'en modsat stubben, som assert'ede på UUT.

Oftest tager mock længere tid at skrive, de er nemme at skrive forkert og sværere at genbruge end stubs. Årsagen til dette er mocks skal have lidt "hukommelse" til at gemme parameter, om metoden blev kaldt og antal gange en metode blev kaldt.

4 NSubstitute

Nsubstitute er et isolations framework, som kan oprette fakes på baggrund af interfaces. I Nsubstitue kan vi asserte på resultatet af et funktionskald (STUBS) eller om en funktion i en mock er blevet kaldt med Recieved. (MOCK)