**Coverlet**

**Prezentare generală**:

Coverlet este un framework de acoperire a codului cross-platform pentru .NET, folosit pentru a măsura cât de mult din codul tău este acoperit de teste unitare. Se integrează bine cu platforma de testare .NET și poate fi utilizat cu framework-uri populare de testare precum NUnit, xUnit și MSTest.

**Caracteristici cheie**:

*Integrare*: Coverlet se integrează perfect în linia de comandă .NET Core și poate fi folosit ca un pachet NuGet, o unealtă .NET, sau ca parte a sarcinilor MSBuild.

*Tipuri de acoperire*: Suportă acoperirea liniilor, a ramurilor și a metodelor, oferindu-ți informații detaliate despre părțile testate ale codului tău.

*Output*: Generează rapoarte detaliate de acoperire, care pot fi ușor interpretate și integrate în fluxul de dezvoltare.

**Stryker**

**Prezentare generală**:

Stryker este o unealtă de testare a mutațiilor pentru .NET și alte limbaje de programare, care modifică în mod deliberat codul sursă pentru a simula erori și pentru a evalua calitatea testelor. Scopul său principal este de a identifica dacă testele tale sunt suficient de robuste pentru a detecta și a preveni erorile introduse în cod.

**Caracteristici cheie**:

*Testare a mutațiilor*: Stryker modifică în mod aleatoriu porțiuni ale codului sursă și rulează testele pentru a vedea dacă acestea detectează și raportează corect schimbările. Dacă testele trec, este posibil să existe deficiențe în acoperirea testelor.

*Generare rapoarte*: Furnizează rapoarte detaliate despre mutații, inclusiv rata de supraviețuire, mutații ucise (“Killed”) și mutații supraviețuitoare (“Survived”). Aceste rapoarte ajută la identificarea zonelor de cod și a testelor care necesită îmbunătățiri.

**Concluzie**

**Coverlet** se concentrează pe măsurarea acoperirii codului de către teste, oferind informații despre părțile testate și ne-testate ale codului.

**Stryker** se concentrează pe evaluarea calității testelor prin simularea erorilor în codul sursă și identificarea dacă testele sunt suficient de robuste pentru a detecta aceste erori.

În general, utilizarea ambelor instrumente în combinație poate îmbunătăți semnificativ calitatea și robustețea codului, asigurându-se că este bine testat și capabil să facă față schimbărilor și erorilor potențiale.