# DEV2 : tester automatiquement ses programmes

[loig.jezequel@univ-nantes.fr](mailto:loig.jezequel@univ-nantes.fr)

# Objectifs du cours

## Tester ses programmes

Quand on programme, on doit tester r´eguli`erement son code, id´ealement `a chaque modification de celui-ci. C¸ a peut ˆetre laborieux.

## Automatiser les tests

Pour ne pas avoir `a sans cesse refaire des tests `a la main on peut

´ecrire et mettre `a jour un jeu de tests qui pourra ˆetre automatiquement utilis´e sur notre programme.

## E´valuer l’efficacit´e du code

Un jeu de test permettra aussi de mesurer le temps de calcul de notre code sur un large ensemble de cas, ce qui permet d’´evaluer son efficacit´e.

# Automatisation des tests en Go

## La base pour tester les paquets d’un module

* La commande *go test*
* Le paquet *testing*

La commande *go test*

* Lit les fichiers se terminant par test.go du dossier courant,
* appelle une `a une toutes les fonctions de test dans ces fichiers,
* affiche le r´esultat de ces tests (r´eussi ou non).

## Fonction de test

* Son nom commence par Test,
* a un unique argument t de type \*testing.T,
* appelle la fonction t.Fail() quand un test est rat´e.

# E´valuation de l’efficacit´e du code

## La base pour ´evaluer l’efficacit´e des paquets d’un module

* La commande *go test -bench=.*
* Le paquet *testing*

La commande *go test -bench=.*

* Lit les fichiers se terminant par test.go du dossier courant,
* appelle toutes les fonctions de benchmarking dans ces fichiers,
* affiche le r´esultat de ces ´evaluations.

## Fonction de benchmarking

* Son nom commence par Benchmark,
* a un unique argument b de type \*testing.B,
* doit utiliser b.N fois le code `a ´evaluer.

Quelques pr´ecisions sur *go test*

## Paquets `a tester

On peut les pr´eciser en listant leurs noms longs `a la suite de la commande. Sinon, la commande travaille uniquement sur le dossier courant.

## Tests et benchmarks

Par d´efaut, quand on fait l’´evaluation de l’efficacit´e du code (*go test -bench=.*) cela ex´ecute aussi les tests. Pour juste faire l’´evaluation on peut pr´eciser la commande :

*go test -run=Bench -bench=.*

## -run= et -bench=

Ces arguments permettent en fait de filtrer les tests `a r´ealiser (run) et les benchmarks `a r´ealiser (bench), on peut mettre n’importe quelle expression rationnelle apr`es le =

# Quelques ressources

* Infos sur la commande *go test* : *go help test*
* Documentation du paquet testing :

<https://pkg.go.dev/testing>