

# 06 – Couverture

partie 1 : le code

Thibaut HENIN

[www.arsouyes.org](http://www.arsouyes.org)

# The QA Team Testing the App

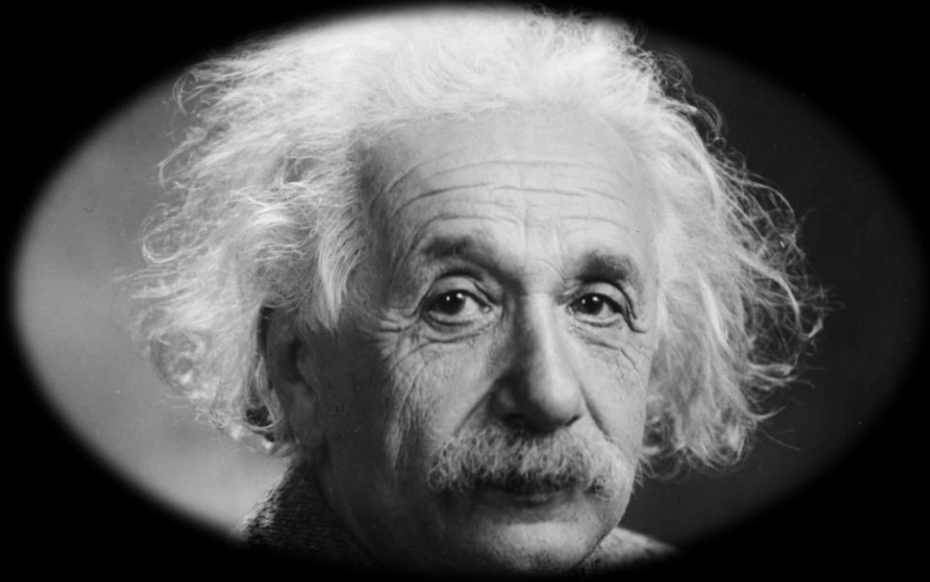
<https://www.youtube.com/watch?v=d63rj2KXp74>



# Quel est le problème ?

S'il n'y a pas de solution, c'est qu'il n'y a pas de problème

Stop ou encore ?



# Aujourd'hui...

## Partir du code source

Au-delà des spécifications

(ce code et ses à côtés)

Tests en boîte blanche

(ne rien laisser de côté)

Architectes / Développeurs

(ils testent leur travail)

# Graphe de flot de Contrôle

Une modélisation et un outil formidable

# Diagramme de flot de contrôle...

Mais qu'est-ce ?

Un graphe orienté

(des case, des flèches)

Schématise l'exécution

(case = instruction, flèche = suite)

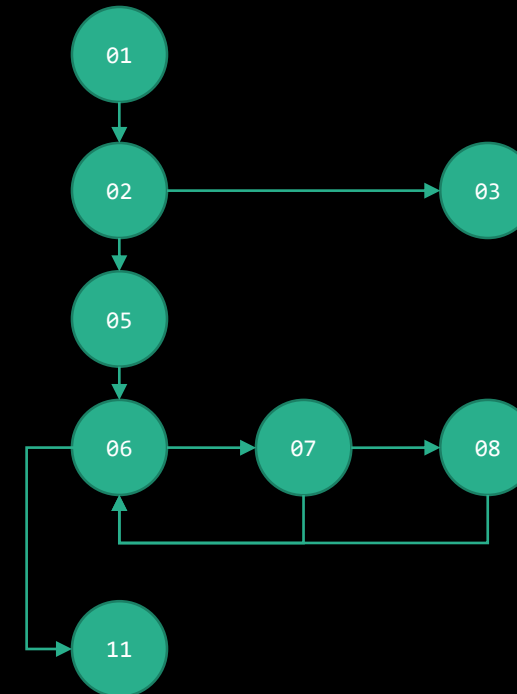
# Exemple de code

```
function countZero($tab) {  
    if (! is_array($tab)) {  
        throw new Exception("Not an array") ;  
    }  
    $result = 0 ;  
    foreach ($tab as $value) {  
        if ($tab === 0) {  
            $result += 1 ;  
        }  
    }  
    return result ;  
}
```



# Diagramme de flot de contrôle

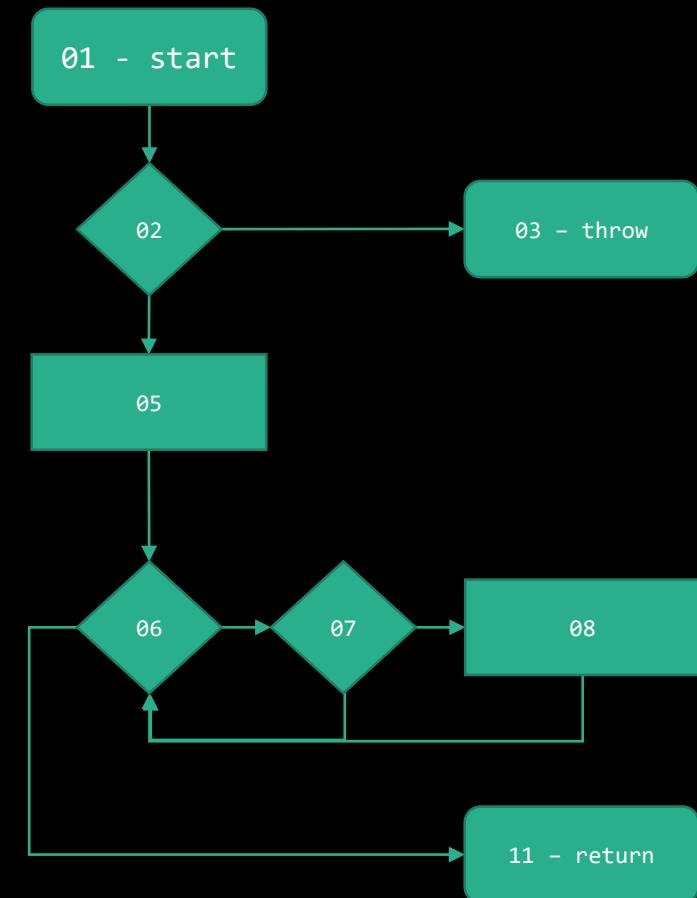
```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ;  
--     }  
05     $result = 0 ;  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ;  
--         }  
--     }  
11     return $result ;  
-- }
```



# Flow Chart

## variante normalisée

```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ;  
--     }  
05     $result = 0 ;  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ;  
--         }  
--     }  
11     return $result ;  
-- }
```



# Modélisation pratique

Au-delà du modèle

Génération automatique

(dans les deux sens)

Algorithmes des graphes

(cycles, chemins, composante connexe, ...)

# Types de couverture

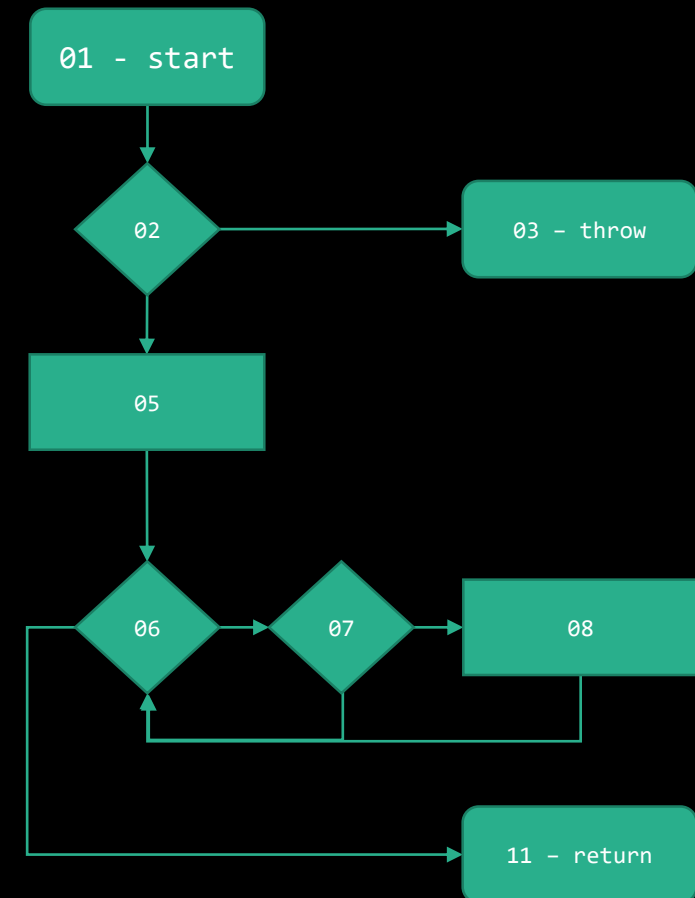
Pour avoir plus ou moins chaud

# Toutes les instructions

## Une fois suffit

```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ;  
--     }  
05     $result = 0 ;  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ;  
--         }  
--     }  
11     return $result ;  
-- }
```

```
$this->assertEquals("1", countZero(0)) ;  
$this->assertEquals("1", countZero([0])) ;
```

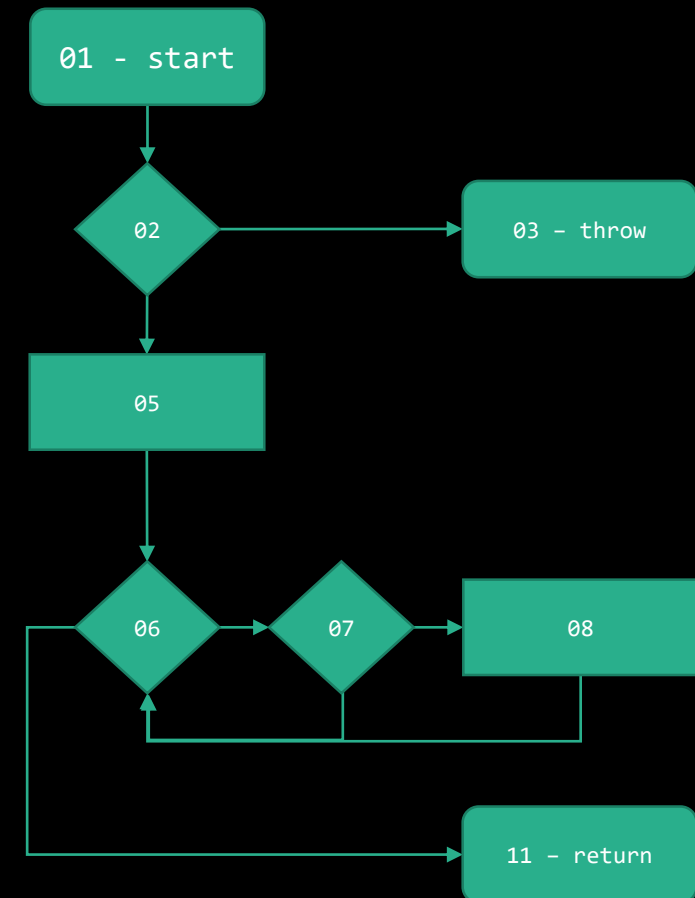


# Toutes les instructions

## Une fois suffit

```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ;  
--     }  
05     $result = 0 ;  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ;  
--         }  
--     }  
11     return $result ;  
-- }
```

```
$this->assertEquals("1", countZero(0)) ;  
$this->assertEquals("1", countZero([0])) ;
```



# Tous les chemins

Impossibles s'il y a des boucles

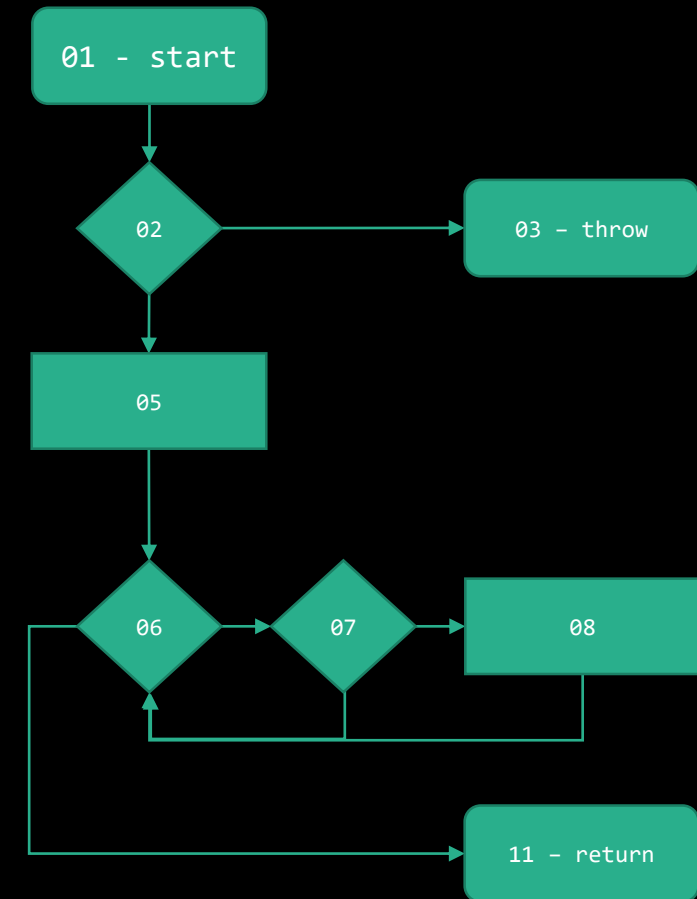
(nombre de cas infinis)

# Toutes les branches

## Une fois de chaque côté

```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ;  
--     }  
05     $result = 0 ;  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ;  
--         }  
--     }  
11     return $result ;  
-- }
```

```
$this->assertEquals("1", countZero(0)) ;  
$this->assertEquals("1", countZero([0])) ;  
$this->assertEquals("1", countZero([0, 1]) ;
```





Complexités

# Complexité cyclomatique

**A Complexity Measure**, McCabe, *IEEE transactions on software engineering*, volume SE-2 issue 4, décembre 1976

Nombre de chemins linéairement indépendants

$$V(g) = e - n + p$$

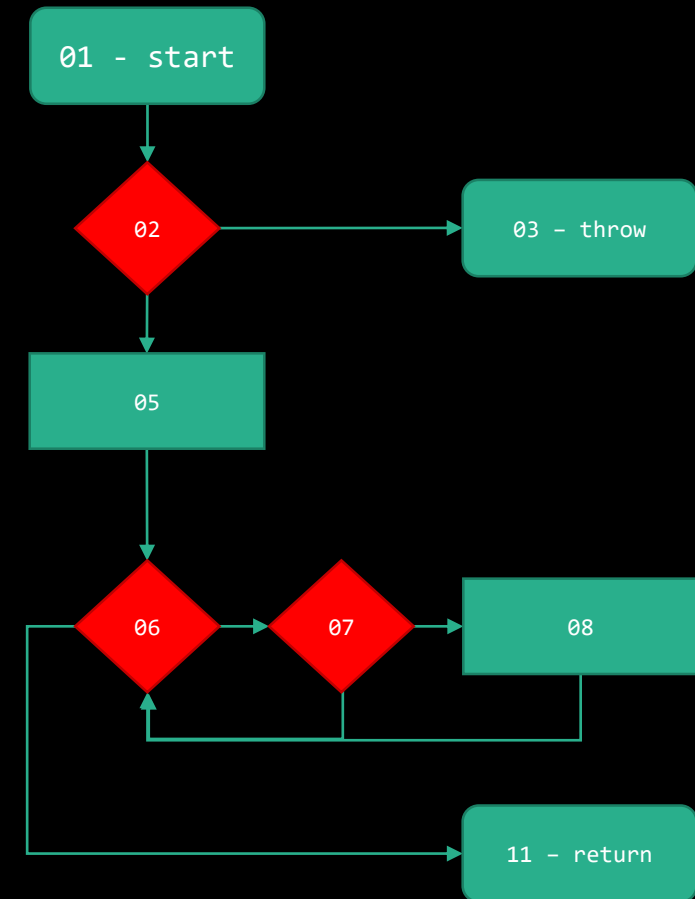
Nombre d'embranchements

(if, select, for, loop, ...)

# Complexité cyclomatique

## Sur l'exemple

```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ;  
--     }  
05     $result = 0 ;  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ;  
--         }  
--     }  
11     return $result ;  
-- }
```



# Complexité NPath

**NPATh: a measure of execution path complexity and its applications**, Brian A. Nejme, *Communications of the ACM*, volume 31, issue 2, février 1988.

Nombre de chemins acycliques

(chemins sans boucles)

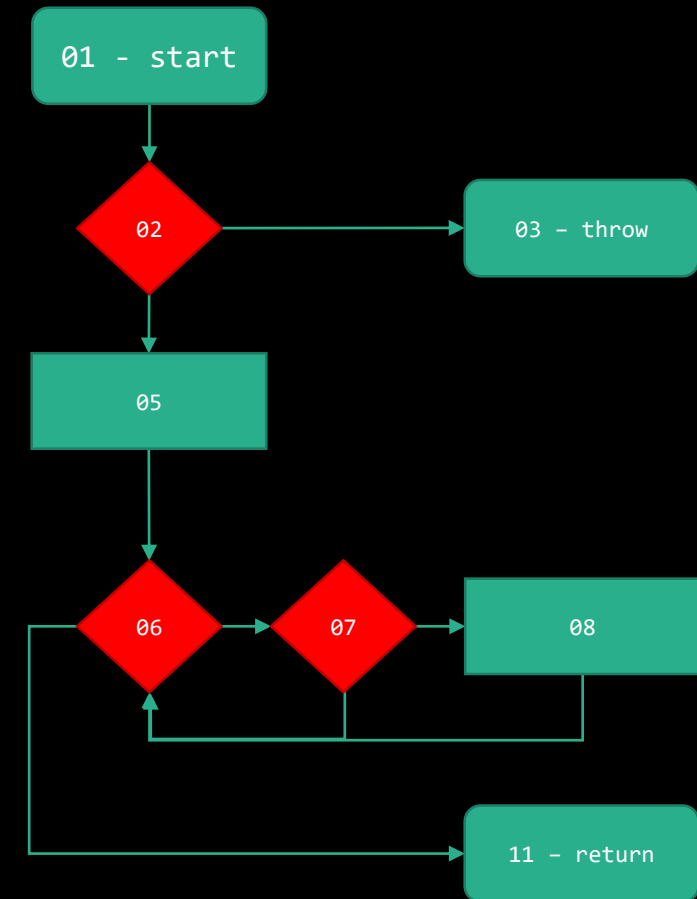
Somme et Produit de choix,

(if, select, for, loop, ...)

# Complexité cyclomatique

## Sur l'exemple

```
01 function countZero($tab) {  
02     if (! is_array($tab)) {  
03         throw new Exception("Not an array") ; // 1  
--     } // ----- +1 = 2  
  
05     $result = 0 ;  
  
06     foreach ($tab as $value) {  
07         if ($tab === 0) {  
08             $result += 1 ; // 1  
--         } // ----- +1 = 2  
  
--     } // ----- +1 = 3  
  
11     return $result ; // bloc_if x bloc_for = 2 x 3 = 6  
-- }
```

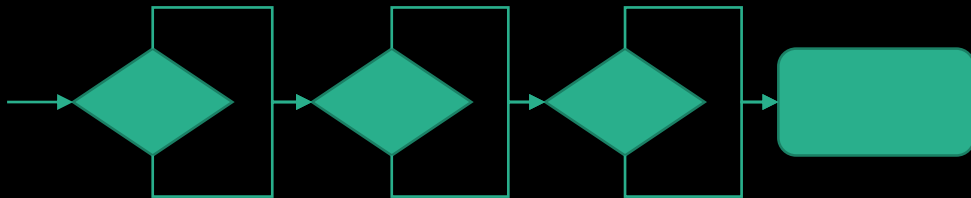


# Liens entre cyclomatique et NPath

$$V(g) \leq NPath(g) \leq 2^{V(g)}$$

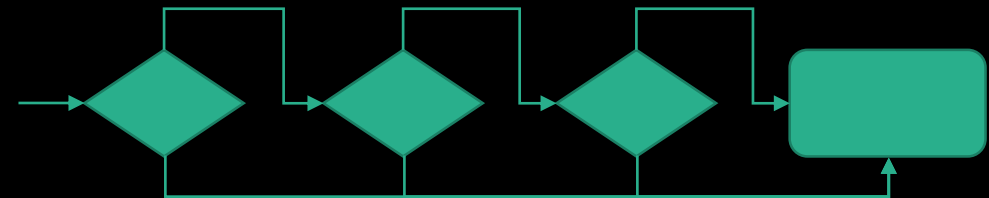
Maximum si séquentielles

(if end ; if end ; ...)



Minimum si imbriqués

(if then if then if ...)



# Rassurez-vous !

**Des outils existent**

(et vous allez en voir plusieurs en TP)

Comment s'en servir



# Seuil de qualité

## Mesure automatique

(i.e. PHPUnit avec `code_coverage`)

## Rejet automatique

(i.e. CI/CD, gitlab, github, ...)

# Indicateur de potentialité

Rapports de couverture

(i.e. HTML ou intégré aux IDEs)

Indique où porter un effort

(peu de test  $\Rightarrow$  plus de risques)

**DELETING A BUNCH OF  
UNUSED BUT TESTED CODE**



**CI FAILS BECAUSE  
AVERAGE COVERAGE GOES DOWN**

