

# Tests de caractérisation : à l'assaut de votre code « Legacy » patrimonial



FÉLIX-ANTOINE  
BOURBONNAIS

B.ING., M.SC, PSM

&

PASCAL ROY

ING., CSM, PSM, PMP



Imaginez du **code patrimonial**  
*(Legacy)*...

Imaginez un outil qui permettrait  
à la fois **d'explorer** ce que le code  
fait **réellement** et de **réusiner** ce  
*vieux code*...



bienvenue



# Félix-Antoine Bourbonnais

B.ing., PSM, M.Sc.



# Pascal Roy

Ing., PSM, CSM, PMP





Formations



Mentorat



Diagnostics



Conférences



Conférenciers

Formateurs

Mentors

Scrum

QA Agile

BDD

Essais automatisés

TDD

Architecture évolutive

DDD

...

Équipe & Affaires

Tech.

Gestion

Conseils stratégiques

Gestion de projets

Agilité

---

# Le code patrimonial


---

Suis-je seul à avoir  
du *Legacy Code* ?





# Qu'est-ce que du *Legacy Code* ?

A photograph of a rusted, vintage car, likely a 1930s model, parked in a field of tall grass. The car is the central focus, showing significant wear and tear. In the background, there are wooden structures and a hillside, suggesting a rural or historical setting. The overall tone is one of age and neglect, which serves as a metaphor for legacy code.

C'est du code difficile à faire évoluer.

Peu importe son âge ou la raison.

# Quelques autres définitions possibles...

- Du code écrit par d'autres
- Du code que plus personne ne veut toucher
- Du code qui n'est plus supporté par ceux qui l'ont écrit
- Du code qui pourrait être réécrit en utilisant de meilleures pratiques de code, d'outils ou de langages
- ...



Du code sans tests

Michael Feathers,  
Working Effectively with Legacy Code



---

# Les effets néfastes du *Legacy Code*

---



# Que faire avec mon code *Legacy* ?

Deux grandes  
approches...

La peur : le pire ennemi du  
développement logiciel



*Tanné* de stresser pour une livraison, de *déboguer*, d'avoir peur de briser?

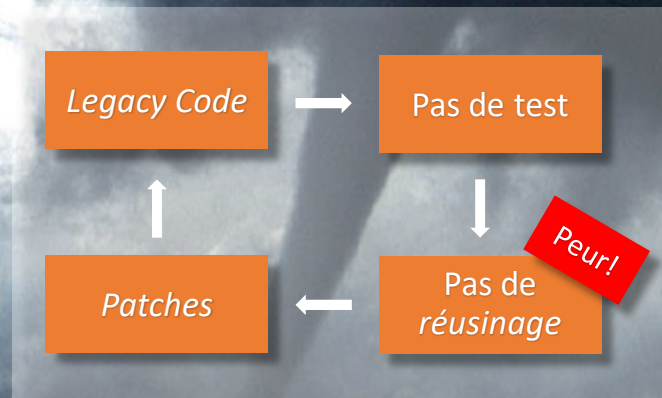




S.v.p. donnez-  
moi un nouveau  
projet !@/\$%!/%



# La descente aux enfers



---

# Stratégie de rénovation

---

Pourquoi ne pas  
**revitaliser**  
votre code?








Bien outillé, vous pouvez rénover !



Graduellement,  
tout en produisant  
de la valeur

A scenic view of a winding road on a hillside at dusk or dawn. The road curves around a hill, and the sky is a mix of soft orange and pale blue. In the distance, a layer of fog or mist fills the valley. A few people are visible on the road and on the hilltop. A green text box is overlaid on the right side of the image.

Sélectionnez votre prochaine  
« Story » et commencez vos  
paiements de dette!

---

Une technique:  
Tests de caractérisation

---

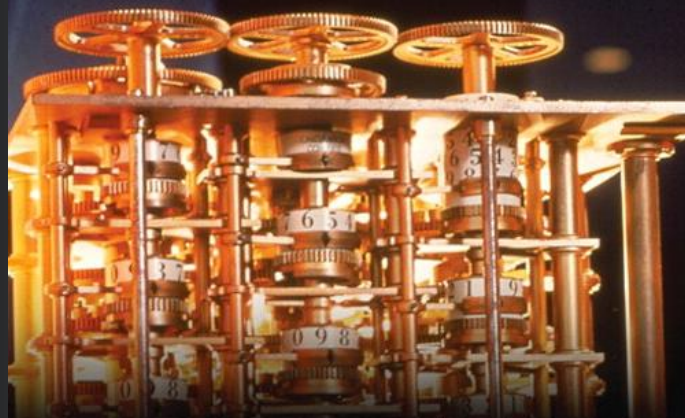




# Test de caractérisation



Robert C. Martin Series



# WORKING EFFECTIVELY WITH **LEGACY CODE**

Michael C. Feathers



Un test de caractérisation est une description du comportement actuel d'un bout de code.

- Michael Feathers



Pour...

C'est un *briseur* de PEUR !

Comprendre et  
documenter ce  
que fait le code

+

Empêcher la  
régression lors  
du *réusinage*

---

# Écriture d'un test de caractérisation

---



# La mécanique d'écriture d'un test de caractérisation

1. **Identifier** et isoler un bout de code à modifier ou à analyser
2. **Écrire** un test qui passe par le bout de code avec une assertion qui échouera
3. **Exécuter** le test et le laisser vous dire quel est le comportement actuel
4. **Changer** votre assertion et le nom du test pour tenir compte du comportement actuel
5. **Répéter...**

# Un exemple simple de code patrimonial?

```
public class SalesUtil {  
    double BQ = 1000.0;  
    double BCR = 0.20;  
    double OQM1 = 1.5;  
    double OQM2 = OQM1 * 2;  
  
    public double calculate(double tSales) {  
        if (tSales <= BQ) {  
            return tSales * BCR;  
        } else if (tSales <= BQ * 2) {  
            return (BQ) * BCR + (tSales - BQ) * BCR * OQM1;  
        } else {  
            return (BQ) * BCR +  
                (tSales - BQ) * BCR * OQM1 +  
                (tSales - BQ * 2) * BCR * OQM2;  
        }  
    }  
}
```

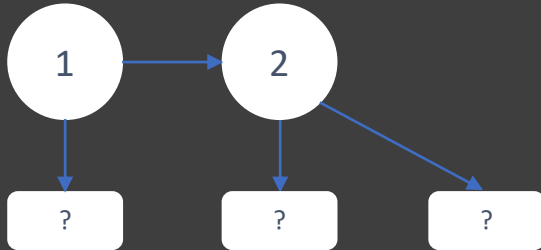
 WTF?

 WTF?

 WTF?

# Étape 1: identifier un bout de code

```
@Test
public void test... {
    assert(...)
}
```

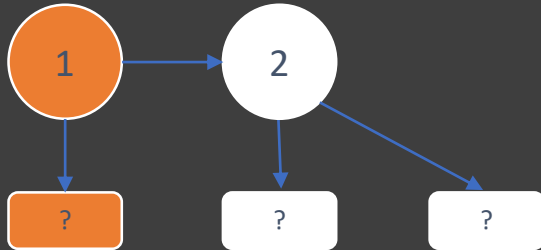


```
public class SalesUtil {
    double BQ = 1000.0;
    double BCR = 0.20;
    double OQM1 = 1.5;
    double OQM2 = OQM1 * 2;

    public double calculate(double tSales){
        if (tSales <= BQ) {
            1 return tSales * BCR;
        } else if (tSales <= BQ * 2) {
            2 return (BQ) * BCR +
                (tSales - BQ) * BCR * OQM1;
        } else {
            return (BQ) * BCR +
                (tSales - BQ) * BCR * OQM1 +
                (tSales - BQ * 2) * BCR * OQM2;
        }
    }
}
```

# Étape 2: écrire une assertion qui ne passe pas

```
@Test
public void testCalculate() {
    assertEquals(
        0.0,
        SalesUtil.calculate(1000.0)
    );
}
```



```
public class SalesUtil {
    double BQ = 1000.0;
    double BCR = 0.20;
    double OQM1 = 1.5;
    double OQM2 = OQM1 * 2;

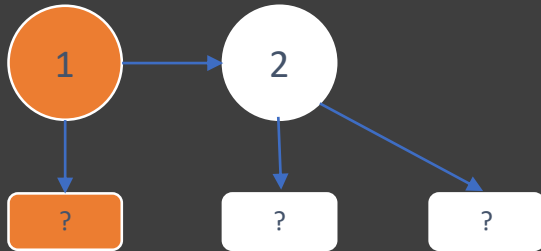
    double calculate(double tSales) {
        if (tSales <= BQ) {
            1 return tSales * BCR;
        } else if (tSales <= BQ * 2) {
            return (BQ) * BCR +
            2 (tSales - BQ) * BCR * OQM1;
        } else {
            return (BQ) * BCR +
                (tSales - BQ) * BCR * OQM1 +
                (tSales - BQ*2)*BCR * OQM2;
        }
    }
}
```





# Étape 3: exécuter le test + trouver le comportement actuel

```
@Test
public void testCalculate() {
    assertEquals(
        0.0,
        SalesUtil.calculate(1000.0)
    );
}
```



```
public class SalesUtil {
    double BQ = 1000.0;
    double BCR = 0.20;
    double OQM1 = 1.5;
    double OQM2 = OQM1 * 2;
```

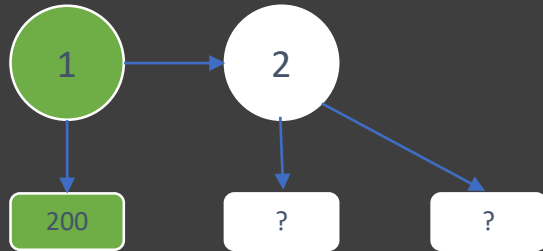
```
double calculate(double tSales) {
    if (tSales <= BQ) {
        return tSales * BCR;
```

```
> junit.framework.AssertionFailedError:
expected:<0.0> but was:<200.0>
```

```
}
}
```

# Étape 4: Remplacer par le comportement découvert

```
@Test
public void lessThanBaseQuota_useBaseCommissionRate()
{
    assertEquals(
        200.0,
        SalesUtil.calculate(1000.0)
    );
}
```



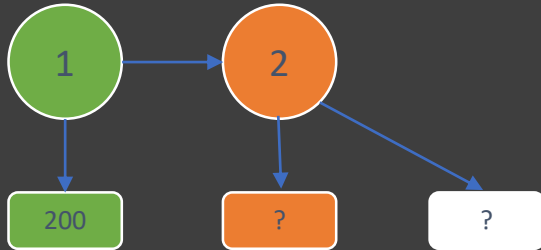
```
public class SalesUtil {
    double BQ = 1000.0;
    double BCR = 0.20;
    double OQM1 = 1.5;
    double OQM2 = OQM1 * 2;

    double calculate(double tSales) {
        if (tSales <= BQ) {
            1 return tSales * BCR;
        } else if (tSales <= BQ * 2) {
            return (BQ) * BCR +
            2 (tSales - BQ) * BCR * OQM1;
        } else {
            return (BQ) * BCR +
                (tSales - BQ) * BCR * OQM1 +
                (tSales - BQ * 2) * BCR * OQM2;
        }
    }
}
```

# Étape 5: Répéter

...

```
@Test
public void testCalculate() {
    assertEquals(
        0.0,
        SalesUtil.calculate(2000.0)
    );
}
```



```
public class SalesUtil {
    double BQ = 1000.0;
    double BCR = 0.20;
    double OQM1 = 1.5;
    double OQM2 = OQM1 * 2;

    double calculate(double tSales) {
        1 if (tSales <= BQ) {
            return tSales * BCR;
        } else if (tSales <= BQ * 2) {
            2 return (BQ) * BCR +
                (tSales - BQ) * BCR * OQM1;
        } else {
            return (BQ) * BCR +
                (tSales - BQ) * BCR * OQM1 +
                (tSales - BQ * 2) * BCR * OQM2;
        }
    }
}
```

Attention aux « tant qu'à y être » !  
Ciblez uniquement ce que vous voulez modifier.



---

# Démonstration

---

On n'a pas le temps  
de faire ça ?!?



Combien de temps ça prend pour  
**comprendre** un bout de code *Legacy*  
avant de le modifier?

---

# Particularités d'un test de caractérisation

---

# How is a CT different?

Test unitaire

Caractérisation

- Spécification du comportement **requis**
- Comportement **connu** et **nouveau** code
- Permanent

- Spécification du comportement **actuel**
- Code **patrimonial**, comportement **flou** ou **perdu**
- Temporaire




Est-ce qu'entourer mon application avec des tests bout-en-bout peut m'aider à caractériser ?



---

Refonte ou réusinage ?

---



Le réusinage (refactoring)  
n'est pas une petite refonte!

Pourquoi les  
gestionnaires/clients/chargés de  
projet ont-ils si peur du réusinage ?





Attention au réusinage en *Big Bang* !



# Refonte ou réusinage ?!?

Refaire

Rénover / Revitaliser

Mais parfois  
inévitable !

Une dépense

Big Bang

Risque très élevé

Pas de nouvelle valeur

Paiements réguliers (dette)

Étape par étape


Risque moindre

Produit de la valeur

# La stratégie

Comme la prise en charge d'un patient dans une urgence...

> on veut **limiter les dommages** et **focaliser** sur l'objectif le plus **pressant**

A close-up photograph of a child's hand holding a small, metallic, insect-like robot. The robot has a spherical head with a small protrusion, a segmented body, and thin, transparent wings. The child's face is blurred in the background, looking intently at the robot. The lighting is dramatic, highlighting the metallic texture of the robot against a dark background.

C'est le **changement à faire** qui guide notre intervention

---

# Conclusion

---

# Maintenant... comment réusiner ?

Cette présentation montre à **comprendre** et **sécuriser** votre *patrimoine*...

Maintenant, vous pouvez apprendre à **rénover** votre *patrimoine*:

- Sprout Methods/Classes
- Instance Delegator
- Extract to Method
- ...



The image shows a large aircraft on a tarmac at sunset. The sky is a vibrant orange and red. Several technicians are silhouetted against the bright background, working on the aircraft. One technician is on the left, another is on the right, and several others are visible underneath the plane. The overall scene conveys a sense of busy, professional maintenance work.

Le défi moderne: la  
maintenabilité



La pourriture du code  
n'est pas une  
« loi naturelle »



Le code patrimonial tue la flamme!

Although our first joy of programming may have been intense, the misery of dealing with legacy code is often sufficient to extinguish that flame.

Michael Feathers,  
Working Effectively with Legacy Code





Le test de caractérisation...  
Ajoutez-le à votre boîte à outils!





La « *patrimonialite* », ça se soigne !



MERCI .



Toutes nos présentations

[conferences.elapsetech.com](https://conferences.elapsetech.com)

Diapositives et références

[conferences.elapsetech.com  
/legacy-tests-characterisation](https://conferences.elapsetech.com/legacy-tests-characterisation)

Félix-Antoine Bourbonnais

Pascal Roy

Site

[elapsetech.com](https://elapsetech.com)

Twitter

[@fbourbonnais](https://twitter.com/fbourbonnais)

Courriel

[fbourbonnais@elapsetech.com](mailto:fbourbonnais@elapsetech.com)

[pascalroy@elapsetech.com](mailto:pascalroy@elapsetech.com)

LinkedIn

[linkedin.com/in/fbourbonnais/fr](https://linkedin.com/in/fbourbonnais/fr)

[ca.linkedin.com/in/roypa](https://ca.linkedin.com/in/roypa)