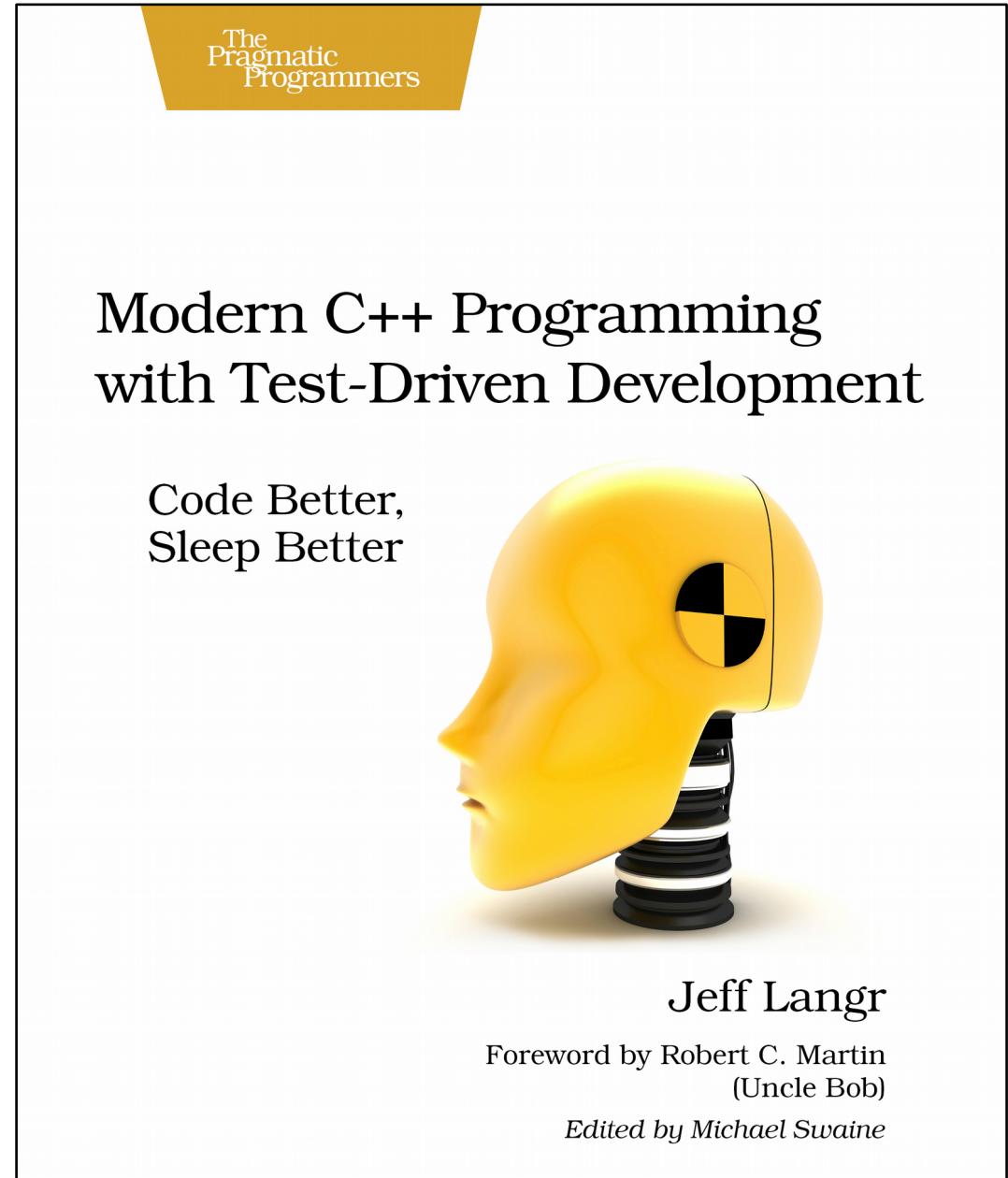


# Testgestuurde ontwikkeling

Richèl Bilderbeek 

# Wat is testgestuurde ontwikkeling?

- Het ontwikkelen van code met de juiste hoeveelheid tests



# Waarom tests?

- Doel:
  - code zonder fouten
  - (heldere architectuur)
  - (gemakkelijk uit te breiden)
  - (snelle ontwikkeling van code)
- Code die niet getest is bevat snel fouten
- Code die niet getest kán worden
  - bevat nog meer fouten
  - (is als een zwart doos)
  - (is niet gemakkelijk uit te breiden)

11/2  
9/9

0800 Autan started  
1000 " stopped - autan ✓ { 1.2700 9.037.847.025  
13' UC (032) MP - NC 1.2700 9.037.846.995 connect  
033 PRO 2 2.130476915  
connect 2.130476915  
Relays 62 in 033 failed special speed test  
1m relay 10ms test.  
Relays changed  
1100 Started Cosine Tape (Sine check)  
1525 Started Multi Adder Test.

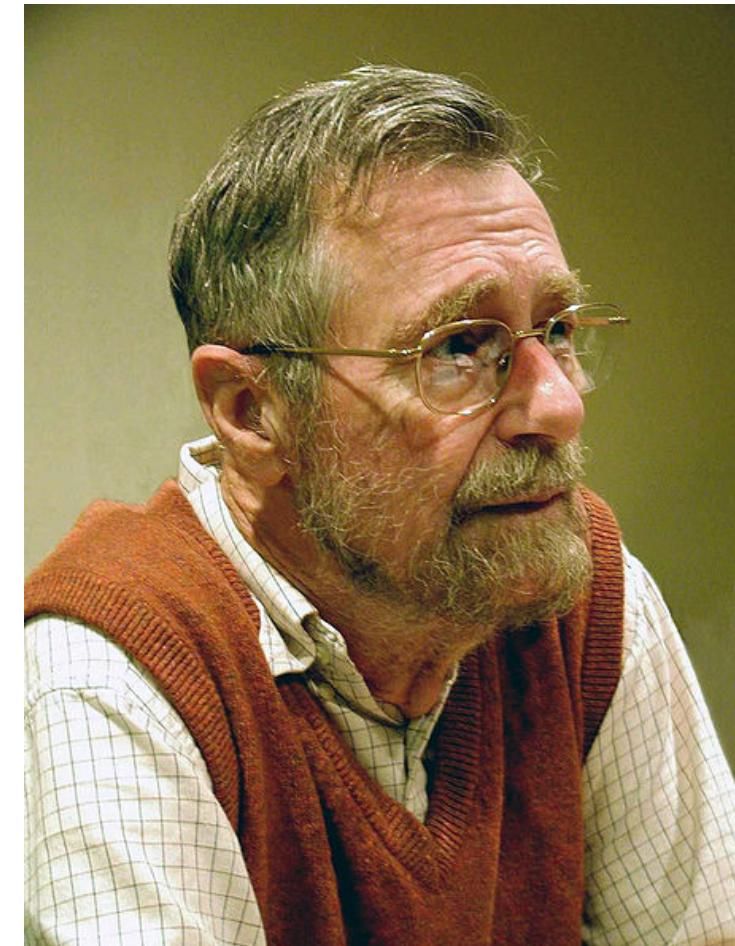
1545 Relay #70 Panel F  
(moth) in relay.

1650 Autan started.  
1700 closed down.

Relay 2145  
Relay 3370

# Hoeveel tests?

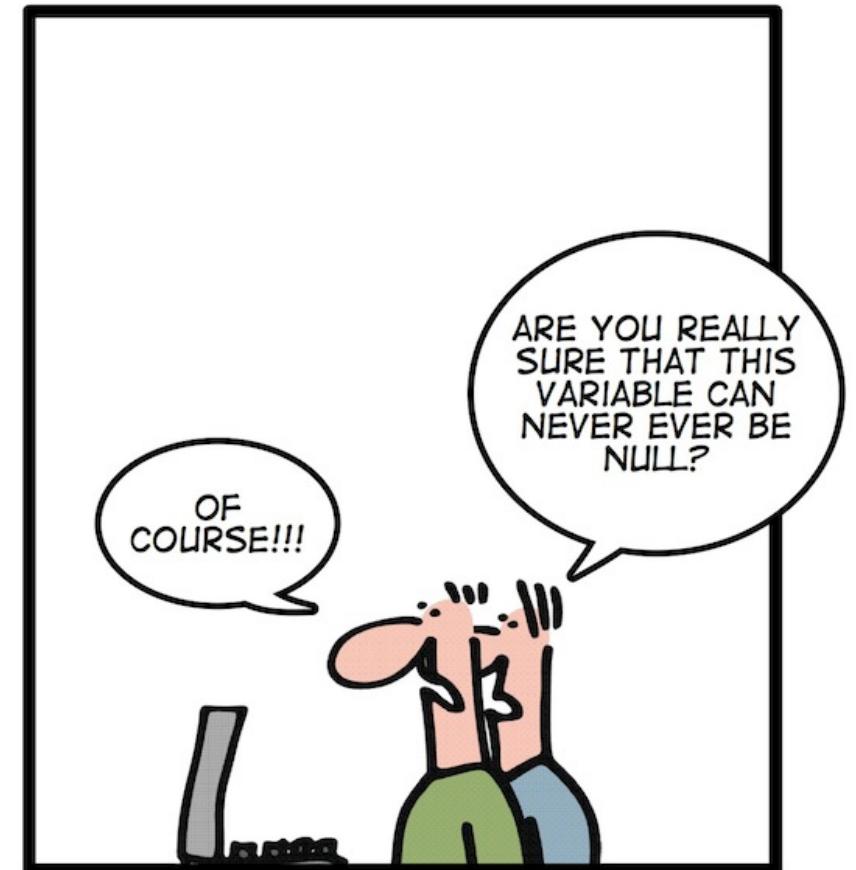
- “Testing can be used to show the presence of errors, never their absence!” Edsger Dijkstra
- Te weinig: veel fouten worden niet ontdekt
- Te veel: veel juistheden worden meerdere malen gecheckt
- Wat is de balans?



# Wat is een fout?

- Logica fout
  - De programmeur maakt een denkfout
  - Bijvoorbeeld: doet een deling door nul
- Runtime fout
  - De hardware loopt tegen een beperking aan
  - Bijvoorbeeld: het geheugen is vol

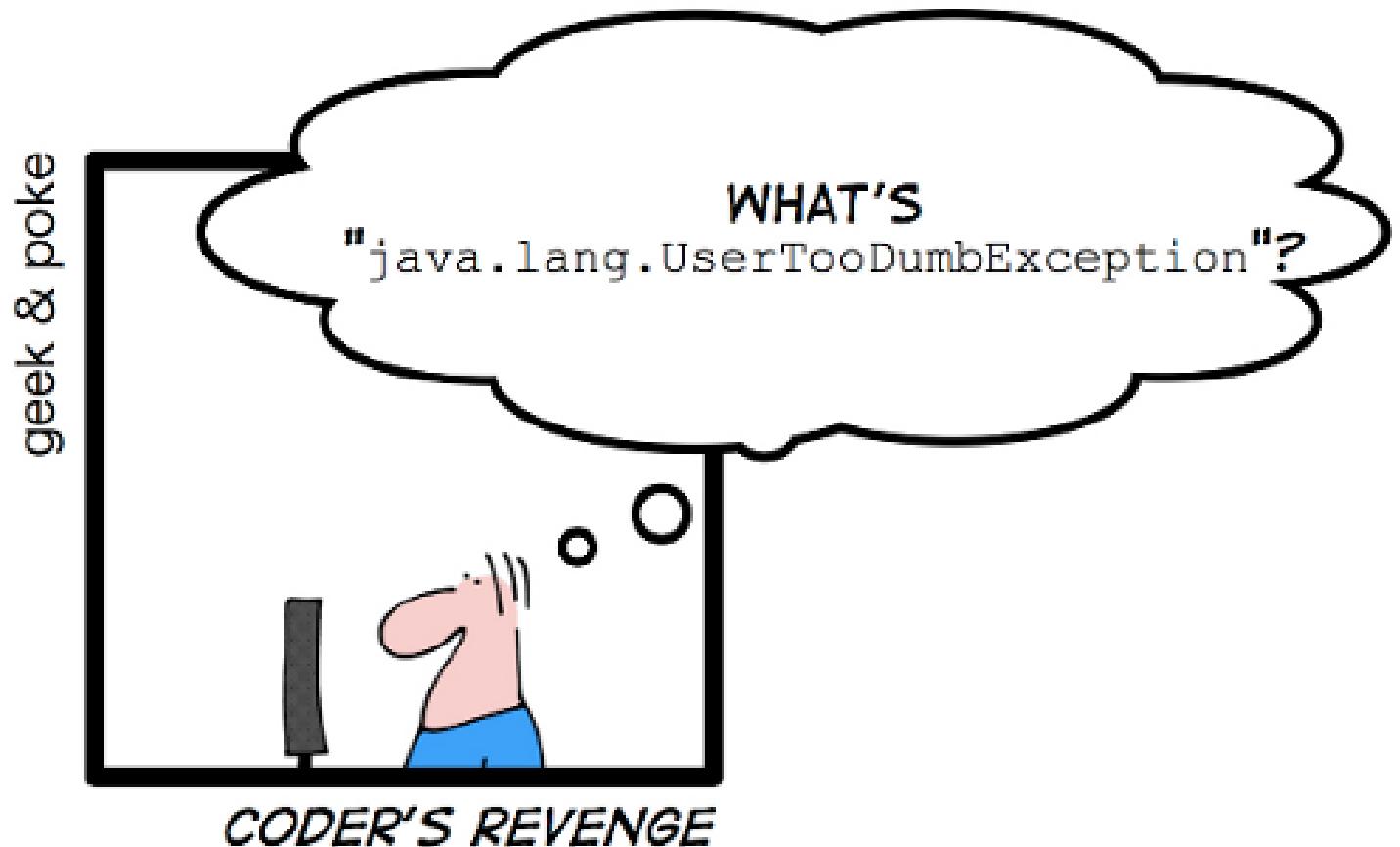
*SIMPLY EXPLAINED*



NullPointerException

# Wat is geen fout?

- De gebruiker geeft onjuiste invoer
- Bijvoorbeeld: een commando dat niet bestaat



# Logica fouten

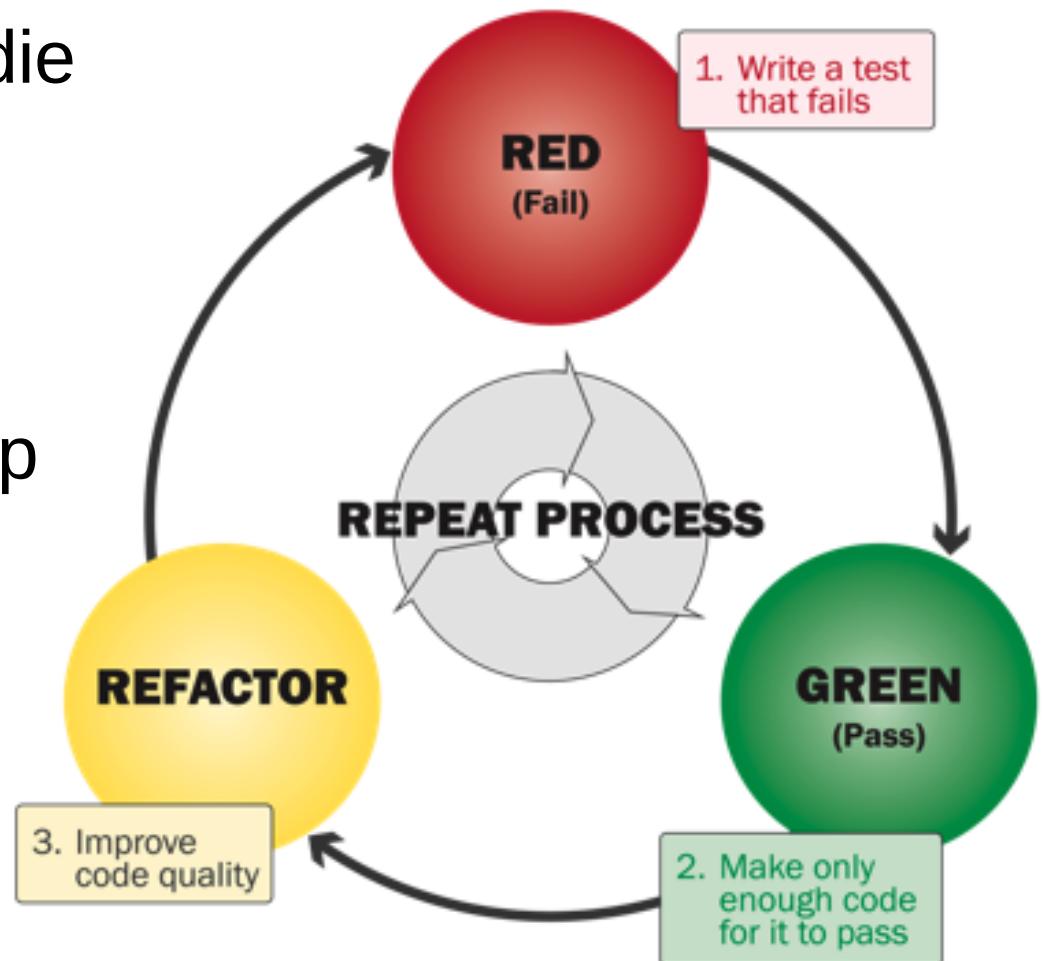
- De programmeur maakt een denkfout
- Voorbeeld: een functie die het onjuiste antwoord geeft
- Test: roep de functie aan, met verschillende argumenten en kijk of eruit komt wat verwacht wordt

# Voorbeeld

```
bool IsPriem(const int getal)
{
    /* berekening */
    /* return true of false */
}
```

# Ontwikkeling IsPriem

- Cyclus testgestuurde ontwikkeling:
  - Rood: Schrijf een test die faalt
  - Groen: De test slaagt
  - Refactor: verbeter de nieuwe code, ruim troep op, check code in



# Ontwikkeling IsPriem

- Zo klein mogelijke stappen
- Een stap tegelijk



# Testomgeving Arduino

```
void setup()
{
    Serial.begin(9600);
    if /* */ {
        Serial.println("Test mislukt!");
    }
}
```

# Rood: schrijf een test die faalt

```
void setup() {  
    Serial.begin(9600);  
    if (!IsPriem(2)) {  
        Serial.println(  
            "Twee moet priem zijn"  
        );  
    }  
}
```

# Groen: de test slaagt

```
bool IsPriem(const int getal) {  
    return true;  
}  
  
void setup() {  
    Serial.begin(9600);  
    if (!IsPriem(2)) {  
        Serial.println("Twee moet priem zijn");  
    }  
}
```

# Refactor

- Inchecken

```
git add --all :/
```

```
git commit -m "IsPriem: 2 is priem"
```

# Rood: schrijf een test die faalt

```
void setup() {  
    /* Eerder code */  
  
    if (IsPriem(4)) {  
  
        Serial.println(  
            "Vier is geen priem"  
        );  
  
    }  
  
}
```

# Groen: de test slaagt

```
bool IsPriem(const int getal) {  
    for (int i=2; i!=getal; ++i)  
    {  
        if (getal % i == 0) return false;  
    }  
    return true;  
}  
  
void setup() { /* */ }
```

# Refactor

- Inchecken

```
git add --all :/
```

```
git commit -m
```

```
"IsPriem: 4 is geen priem"
```

# Rood: een test die faalt

```
void setup() {  
    /* Eerder code */  
    if (IsPriem(1)) {  
        Serial.println(  
            "Een is geen priem"  
        );  
    }  
}
```

# Groen: de test slaagt

```
bool IsPriem(const int x) {  
    if (x == 1) return false;  
    /* rest code */  
}
```

# Refactor

- Inchecken

```
git add --all :/
```

```
git commit -m
```

```
"IsPriem: 1 is geen priem"
```

# Rood: een test die faalt

```
void setup() {  
    /* Eerder code */  
    if (IsPriem(0)) {  
        Serial.println(  
            "Nul is geen priem"  
        );  
    }  
}
```

# Groen: de test slaagt

```
bool IsPriem(const int x) {  
    if (x <= 1) return false;  
    /* rest code */  
}
```

# Refactor

- Inchecken

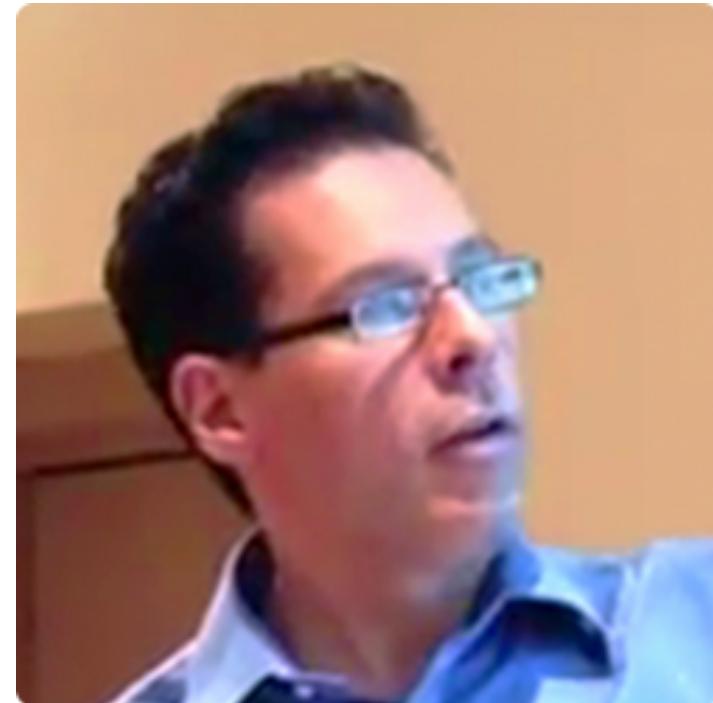
```
git add --all :/
```

```
git commit -m
```

“IsPriem: nul en lager geen priem”

# Snelheid

- “It is far, far easier to make a correct program fast than it is to make a fast program correct.”  
Herb Sutter
- De snelheid van een functie is te meten
- De snelheid van een (mogelijk) verbeterde functie zou hoger moeten zijn
- De verbeterde functie moet dezelfde antwoorden teruggeven



# Mogelijk eindresultaat

```
bool IsPriem(const int x) {  
    if (x <= 1) return false;  
    const int max = static_cast<int>(  
        sqrt(static_cast<double>(x))  
    ) + 1  
};  
for (int i=2; i!=max; ++i) {  
    if (x % i == 0) return false;  
}  
return true;  
}
```

# Conclusie

- Testgestuurde ontwikkeling
  - heeft een systematische werkwijze
  - levert code op die testbaar is
  - levert code op die getest is juist te zijn
  - test niet meer dan nodig
  - maakt snelle ontwikkeling mogelijk