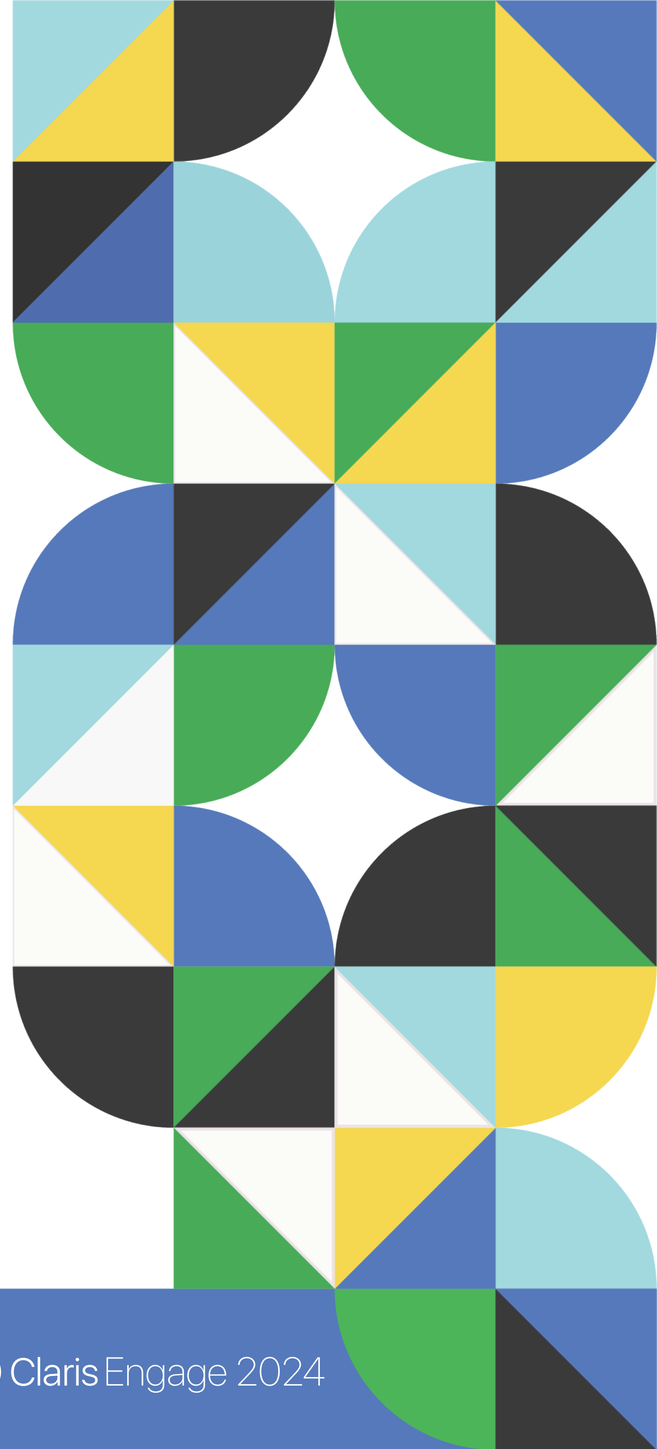
Bearskin Hats

Using Guard Clauses for More Robust Programming

February 8, 3:00 pm–4:00pm

Cristos Lianides-Chin
Codence



Who Am I?

Platform Architect at Codence

Tech since 2005, FileMaker since 2012

Passionate about ~~lazy programming~~ **efficient coding practices**

cristos.lianides-chin@codence.com

codence

Who is This Talk For?

01

Beginner Developers

- Gain a foundational understanding of guard clauses and their role in simplifying code.
- Learn the basics of clean coding practices and how to start implementing them in your scripts.

02

Intermediate Developers

- Enhance code readability and maintainability.
- Learn to apply guard clauses more effectively, especially in complex scripting scenarios.

03

Tech Leads

- Define & implement coding best practices.
- Guide and mentor team members in writing more efficient and maintainable code.

What We'll Cover

✅ In Scope

Guard Clauses

Single-Pass Loops

(If time) Guarding Scripts vs Calcs

❌ NOT In Scope

Error Handling

Transactions

Guard Clauses

01

Avoid the Arrow Pattern

If vs Guard

02

Fail Fast

Skip to the end when something goes wrong

03

Further Reading

- https://www.codementor.io/@clintwinter/use-guard-clauses-for-cleaner-code-1rrsczgwxp
- https://en.wikipedia.org/wiki/Guard_(computer_science)

[DEMO]
If vs Guard

"It is not enough for code to work. It must also be clean, readable, and understandable."

~ Robert C. Martin, *Clean Code*

[DEMO]
Exit Script
v Exit Loop If

Try-Catch

Especially Useful in Iterators

```
data = ["10", "20", "a", "30", "40b", "50"]

for item in data:
    try:
        number = int(item)  # Attempt to convert string to integer
        print(f"Processed number: {number}”)

    except ValueError:
        print(f"Invalid item encountered: '{item}' - Skipping")
```

```
Processed number: 10
Processed number: 20
Invalid item encountered: 'a' - Skipping
Processed number: 30
Invalid item encountered: '40b' - Skipping
Processed number: 50
```

Single-Pass Loops

Try-Catch in FileMaker

```
# TRY
Loop
   # GUARD
   Exit Loop If [ x.ifThenThrowException ( value.isNullOrError ( $listSystemTables ) ; "This script requires a l… ]

   # FOREACH ( $tableName in $listSystemTables )
   Loop
      Exit Loop If [ Let ( [  $i1Max = ValueCount ( $listSystemTables ) ;$i1 = $i1 + 1 ;$i10 = $i1 - 1 ]; If (  $… ]
      Set Variable [ $tableName ; Value: GetValue ( $listSystemTables ; $i1 ) ]

      # INNER TRY
      Loop

         Go to Layout [ $tableName ; Animation: None ]

         # GUARD
         Exit Loop If [ x.ifThenThrowException ( not ( Get ( LayoutTableName ) = $tableName ) ; "Expected to be o… ]
         Exit Loop If [ Get ( TotalRecordCount ) > 0 // SAFE EXIT: Records exist on this table, no need to contin… ]

         New Record/Request

         Exit Loop If [ True //END INNER TRY ]
      End Loop
      # END INNER TRY

   End Loop
   # END FOREACH ( $tableName in $listSystemTables )

   Exit Loop If [ True //END TRY ]
End Loop

# CATCH
If [ x.countExceptions ]
   Set Variable [ $~void ; Value: script.response.addExceptions ]
End If

# FINALLY

Exit Script [ Text Result: script.response ]
```

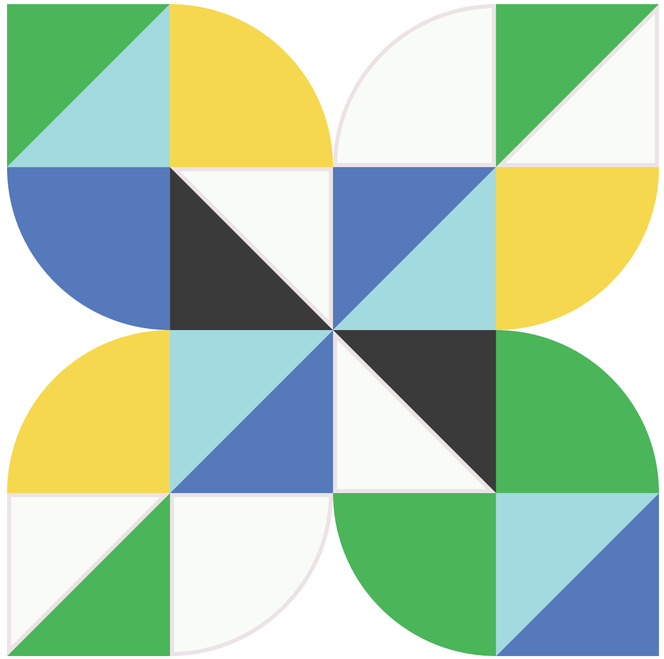
[DEMO] Try-Catch in an Iterating Loop

Scripts vs Calculations

Can’t Try-Catch in Calc Engine

```
// Before we do something slow/expensive, let's set a guard clause with an error message.
isExit = Case ( isExit ; True
;   Length ($1) and Length ($2) ; True // If both $1 and $2 are empty, we don't need to do the expensive stuff
) ;
~errorMessage = Case ( Length ( ~errorMessage ) ; ~errorMessage
;   isExit ; "Error: " & $1 & " " & $2 & " cannot both be set."
);

$3 = If ( not isExit ;
"Do some hard stuff" // Do something slow, like iterating over a list of 10^6 items
) ;
```



Q&A

Thank you.

cristos.lianides-chin@codence.com