

Desire to be proud of what we do

SOFTWARE CRAFTSMANSHIP



PREVENT
CODE ROT



LANGLEY AND
TEAM'S MINDSET

HAVE TO DO IT

WRITE BROTHERS
MINDSET

WANT TO DO IT

**IN 1982, A THEORY WAS
INTRODUCED BY
SOCIAL SCIENTISTS**

JAMES Q. WILSON

WINDOW



LITTER



GRAFFITI



STRUCTURE



SENSE OF ABANDONMENT
BECOMES REALITY



NEGLECTION **ACCELERATES** THE ROT FASTER
THAN ANY OTHER FACTOR.

HOW IS THIS RELATED WITH THE SOFTWARE?

Are the next few questions
sounds familiar?

CHANGES ARE **HARD** TO DO?

SIMPLE CHANGE **IMPACTS**
NUMEROUS MODULES?

IMPLEMENTING SIMPLE
CHANGE TAKES **FOREVER?**

CHANGE IN ONE PLACE
HARMS SOMEWHERE
COMPLETELY UNEXPECTED
AREA?

FIXING A BUG **CAUSES** <N>
MORE?

MODULES ARE
NOT REUSABLE
BECAUSE OF THEIR
DEPENDENCIES?

IT'S EASIER
TO DO **HACKS**
THAN BY THE BOOK?

BINGO!



```
public static String testableH  
    PageData pageData,  
    boolean includeSuiteSetup  
) throws Exception {  
    WikiPage wikiPage = pageData.  
    StringBuffer buffer = new S  
    if (pageData.hasAttribute(")  
        if (includeSuiteSetup) {  
            WikiPage suiteSetup =  
                PageCrawlerImpl.getIn  
                SuiteResponde  
        };  
        if (suiteSetup != null)  
            WikiPagePath pagePath  
            suiteSetup.getPageC  
            String pagePathName =
```



```
public List<ir  
    List<int[]>  
    for (int[] x  
        if (x[0] =  
            list1.ad  
    return list1  
}
```



```
int a = 1;  
if (0 == 1  
    a = 01;  
else  
    1 = 01;
```



```
/** The child Containers belongi  
protected HashMap children = new  
/** The processor delay for this  
protected int backgroundProcesso  
/** The lifecycle event support  
protected LifecycleSupport lifec  
/** The container event listener  
protected ArrayList listeners =  
/** The Loader implementation wi  
protected Loader loader = null;
```

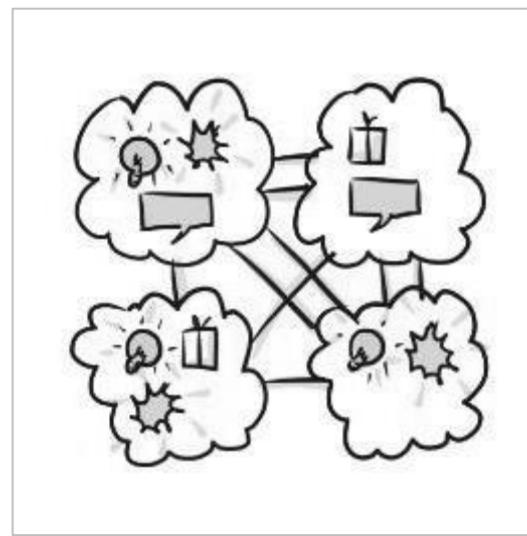


CODE ROT HAS BEGUN?

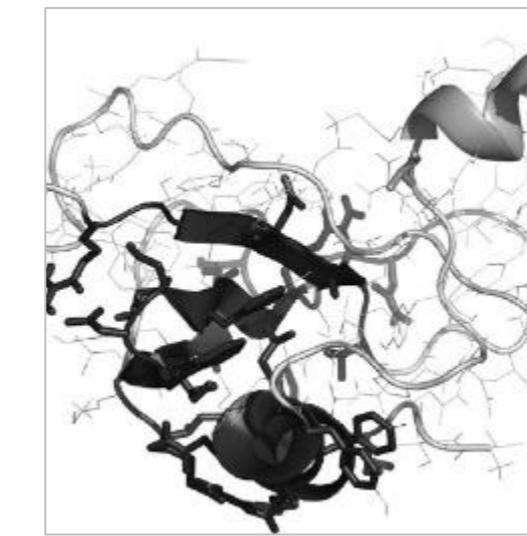
SOFTWARE EROSION



SOFTWARE DECAY



SOFTWARE ENTROPY



SOFTWARE ROT

```
s function name better
.swapcase; end

F=0,G='',H='';_=>+>F<E.length||E.push(0),
{[F]=B.charCodeAt(D++),'[':T=>{if(!E[F])for(T=C]&&—T});C<A.length;++C)H[A[C]]());

#get divs
divs=lambda num: [x for x in ran
+thing) + " "
thing] + " "
forget 2 use recursion
//else w/nested ternaries!
for(var i=maxd-
```

TECHNICAL DEBT

WE KNOW
IT BY MANY
NAMES

HOW CAN WE **STOP** IT?



KEEP THE
CODE CLEAN
AT ALL TIMES

WHAT'S THE CLEAN CODE?

...DOES ONE THING WELL

-- Bjarne Stroustrup

Inventor of C++

...READS LIKE WELL-WRITTEN **PROSE**

-- Grady Booch

Inventor of UML

...IT WAS WRITTEN BY SOMEONE
WHO CARED

-- Michael Feathers

Author of Working effectively with Legacy Code

...FEW PROGRAMMERS KNOW
HOW TO WRITE CODE THAT
HUMAN UNDERSTAND

-- Martin Fowler

Author of Refactoring

...IS WHEN EACH METHOD YOU READ
TURNS OUT TO BE PRETTY MUCH
WHAT YOU EXPECTED

-- Ward Cunningham

*Inventor of Wiki
Coinventor of Extreme Programming*



OUR GOAL IS
NOT TO CODE
BUT TO
COMMUNICATE

1

WHILE MAKING
CHANGES
TO EXISTING CODE

2

WHILE ADDING
NEW CODE

1

WHILE MAKING
CHANGES
TO EXISTING CODE

2

WHILE ADDING
NEW CODE

ANY IDEA WHEN
EXISTING CODE
BECOMES **LEGACY?**

THE MOMENT WE
STOP REFACTORING
OR **STOP CARING**

**MAINTENANCE START THE MOMENT
WE WRITE THE CODE**

WHAT HAPPENS WHEN NOT CARED

```

CREATE PROCEDURE dbo.prv_BundleFilter_set_old
--valide
--date
--idbundle INT = NULL
--idLocalizationSG INT = NULL
--idFOCode INT = NULL
--isMigrated BIT = NULL
--isNCRStatus INT = NULL
--idBilateralStructure INT = NULL
--isInactive INT = 1
--idProfile INT = NULL
AS
DECLARE @toCodesTable TABLE
(
    idFrontOfficeCode INT
)
INSERT INTO #toCodesTable
SELECT *
FROM fn_GetDistinctFOCodes(@idProfile)

DECLARE @CombinedFacility TABLE
(
    idBundle INT NULL INDEX IX_idbundle,
    idMultiFacilityHeader INT NULL ,
    isInactive BIT
)
--Optimisation
SELECT *
INTO #bundle1
FROM sho.txs.Bundle_create()

----- Bundle SELECT -----
----- IF (@idBundle IS NOT NULL )
----- BEGIN
-----     -- step 1 :get all child bundles
-----     INSERT INTO #bundle1
-----     (
        idBundle ,
        idBundleUp ,
        lft ,
        rgt ,
        idTree ,
        valueDate ,
        asOfDate ,
        idItemtype
    )
-----     SELECT DISTINCT
-----         idBundle ,
-----         idBundleUp ,
-----         lft ,
-----         rgt ,
-----         idTree ,
-----         valueDate ,
-----         asOfDate ,
-----         idItemtype
-----     FROM Bundle arc
-----     INNER JOIN Bundle AS B ON B.idTree = arc.idTree
-----     AND B.lft BETWEEN *
-----     WHERE arc.idBundle = @idBundle --> arc Definition

----- step 2 : remove boxes belong to unwanted facilities which are
--Remove un-authorized Boxes
DELETE FROM #bundle1
WHERE idBundle IN (
--Get idBundle for Authorized boxes
SELECT b2.idBundle
FROM dbo.Bundle b2
WHERE b2.idBundle NOT IN (
    SELECT F.idbundle
    FROM Facility F
    INNER JOIN FacilityComp FC ON FC.idFacility
    INNER JOIN #toCodesTable fc ON fc.idFOCode
    WHERE FC.idFOCode = ISNULL(@idFOCode, FC.id)
    AND F.idlocalBL = ISNULL(@localizationSG, F.idlocalBL)
)
)
----- AND b2.idItemType = 3 --Box
)
-----SELECT * FROM #bundle1

----- step 3 : only keep facilities in the scope (remove unwanted fa
DELETE #bundle1
WHERE idBundle NOT IN (
    SELECT F.idbundle
    FROM Facility F
    INNER JOIN FacilityComp FC ON FC.idFacility
    INNER JOIN #toCodesTable fc ON fc.idFrontOfficeCode
    WHERE FC.idFOCode = ISNULL(@idFOCode, FC.idFOCode)
    AND F.idlocalBL = ISNULL(@localizationSG, F.i
)
----- AND #bundle1.idItemType = 2 --Facility
-----SELECT * FROM #bundle1
----- step 4 : remove invalid box bundle
DELETE b
FROM #bundle1 AS b --> on prend les Bundles de BOX uniques
INNER JOIN Facility AS f ON f.idBundle = b.idBundleUp
AND ( isInactive IS NULL
      OR ( isInactive = 1
            AND ( idFaci
                  OR ( f.i
                        AND f.r
                )
            )
      )
      OR ( isInactive = 0
            AND ( idFaci
                  AND f.r
                )
            )
      )
)
----- qui ne sont pas valides.
WHERE b.idBundle NOT IN (
    SELECT idBundle
    FROM Box AS Bc With (INDEX(IX
    WHERE Bc.startDate < @valueDate
    AND ( valueDate < Bc.endDate
          OR Bc.endDate IS Nu
        )
        AND ( isInactive IS NULL
              OR bo.isInactive = 0
            )
        )
)
----- step 5 : Remove DAF template
DELETE b
FROM #bundle1 AS b
INNER JOIN box ON b.idBundle = bcs.idBundle,
INNER JOIN [Transaction] Tr ON bcs.idBox = Tr.idBox
INNER JOIN [TransactionVersion] ON Tr.idTransaction
WHERE Tr.idTransactionStatus = 4

----- step 6 : remove Combinedfacility that don't have any facility
--Remove un-authorized Multi facilities
----- --SELECT * FROM #bundle1
DELETE #bundle1
WHERE idBundle NOT IN (
    -- Get idBundle of MultiFacilities which have authorised Fa
    SELECT Hed.idBundle
    FROM dbo.BMOMultiFacilityHeader hed
    INNER JOIN dbo.BMOMultiFacilityDetail Det ON t
        --AND II
        AND det
    INNER JOIN dbo.Facility F ON det.idFacility =
    INNER JOIN FacilityComp FC ON FC.idFacility =
    INNER JOIN #toCodesTable fc ON fc.idFrontOfficeCode
    WHERE FC.idFOCode = ISNULL(@idFOCode, FC.idFOCode)
    AND F.idlocalBL = ISNULL(@localizationSG, F.i
)
----- AND #bundle1.idItemType = 17 --CombinedFacility
)
-----INSERT INTO #bundle1
----- (
    idbundle ,
    idbundleup ,
    lft ,
    rgt ,
    idtree ,
    valueDate ,
    asofDate ,
    type ,
    idlocalBL ,
    locatortype ,
    idfrontOfficeCode ,
    FrontOfficeCode
)
-----SELECT DISTINCT
----- B.idbundle ,
----- B.idbundleup ,
----- B.lft ,
----- B.rgt ,
----- B.idtree ,
----- B.valueDate ,
----- B.asofDate ,
----- Box ,
----- src.idlocalBL ,
----- src.idbundle ,
----- src.idFrontOfficeCode ,
----- src.FrontOfficeCode
----- FROM Bundle AS B
----- INNER JOIN Box AS Bc ON Bc.idbundle = B.idbundle
----- LEFT JOIN [Transaction] Tr ON Bc.idBox = Tr.id
----- LEFT JOIN [TransactionVersion] ON Tr.idTransac
----- WHERE Bc.startDate < @valueDate
----- AND ( valueDate < Bc.endDate
        OR Bc.endDate IS NULL
        )
        AND ( isInactive IS NULL
              OR bo.isInactive = 0
            )
        AND ( Tr.idTransactionStatus IS NULL
              OR Tr.idTransactionStatus != 4
            )
        --and src.type = 'Facility'
)
-----*****Remove Combinedfacility based on Active status****
INSERT INTO #CombinedFacility
SELECT DISTINCT
    m.idbundle ,
    m.idMultiFacilityHeader ,
    0
FROM #bundle1 AS arc
INNER JOIN BMOMultiFacilityHeader m ON arc.idBundle =
-----UPDATE #CombinedFacility
SET isactive = 1
WHERE idMultiFacilityHeader IN (
    SELECT idMultiFacilityHeader
    FROM dbo.Facility F
    INNER JOIN dbo.BMOMultiFacilityDetail BBD ON BBD.idFacility
    INNER JOIN #CombinedFacility CF ON BBD.idBMOMultiFacil
    WHERE ( F.idFacilityState
        OR ( idFacilityState = 0
              AND F.stateDate > @valueDate
            )
        )
        --stateDate is important
        AND ( BBD.stateDate < @valueDate
              AND ( BBD.endDate < @valueDate
                    OR BBD.endDate IS NULL
                  )
            )
)
-----IF @isInactive IS NOT NULL
DELETE FROM #bundle1
WHERE idbundle IN (
    SELECT idbundle
    FROM #CombinedFacility
    WHERE isactive < @isInactive
)
-----Custom
-----on set le type CombinedFacility
UPDATE #bundle1
SET type = 'CombinedFacility'
idFaci = 1
idlocalBL = idlocalBL
calisnodey = ISNULL(CF.Currency, C.calixdev),
label = m.label
minNCR=ISNULL(CF.minCoverageRatio,100) --Added for combined f
----- INNER JOIN BMOMultiFacilityHeader m ON m.idbundle = B.idbundle
----- INNER JOIN dbo.Customer C ON C.idbundle = B.idbundleUp
----- LEFT OUTER JOIN CombinedFacilityInfo CF ON CF.idBMOMultiFacili
----- CF.startDate < @valueDate
----- AND ( valueDate < CF.endDate
        OR CF.endDate IS NULL
      )
)
-----UPDATE #bundle1
SET calisnodevCOO = ISNULL(CF.Currency, C.calixdev)
FROM #bundle1 B
----- INNER JOIN BMOMultiFacilityHeader m ON m.idbundle = B.idbundle
----- INNER JOIN dbo.Customer C ON C.idbundle = B.idbundleUp
----- LEFT OUTER JOIN CombinedFacilityInfo CF ON CF.idBMOMultiFacili
----- CF.startDate < @valueDate
----- AND ( valueDate < CF.endDate
        OR CF.endDate IS NULL
      )
)
-----UPDATE #bundle1
SET calisnodevCOO = ISNULL(calixdevCOO, calisnodev)
WHERE idItemType = 17
)
-----UPDATE #bundle1
SET #Customer = 'Customer'
iditemtype = 1 -- Ref ItemType
label = CONVERT(VARCHAR(50), C.idParty),
label = BDR.name
calisnodev = C.calixdev
idlocalBL = B.idlocalBL
Customer AS B
----- INNER JOIN Ref BdrTiers AS BDR ON C.idParty = BDR.idParty
WHERE C.idbundle = B.idbundle
IF @@ROWCOUNT = 0
PRINT 'Warning: prv_Bundlefilter_set : no CUSTOMER updated'
----- Facilities
-----UPDATE #bundle1
SET type = 'Facility'
iditemtype = 2 -- Ref ItemType
idkey = F.idkey
label = F.label
calisnodev = C.calixdev,
isMigrated = FC.isMigrated,
isNCRStatus = FC.isNCRStatus ,
idlocalBL = F.idlocalBL ,
localBL = B.localBL
idfrontOfficeCode = F.idFrontOfficeCode ,
FrontOfficeCode = F0.id
isMarginCallAlertScope = FC.isMarginCallAlertScope ,
isMarginCallAlertType = FC.idMarginCallAlertType ,
isNCRAlertScope = FC.idNCRAlertScope ,
isNCRAlertType = FC.idNCRAlertType ,
idBilateralStructure = FC.idBilateralStructure ,
sharedBL = ISNULL(BBL.idShared, 0)
-----#bundle1 AS B
----- INNER JOIN Facility F ON B.idbundle = F.idbundle
----- INNER JOIN Ref LocalBL AS Bl ON F.idlocalBL = Bl.idlocalBL
----- LEFT OUTER JOIN FacilityComp AC ON F.idFacility = AC.idFacility
----- LEFT OUTER JOIN Ref FrontOfficeCode FO ON FC.idFOCode = FO.id
----- LEFT JOIN BMOMultiFacilityHeader BMH ON B.idbundle = F.idbundle
----- ( F.idFacilityState = 0
        AND F.stateDate > @valueDate
      )
        AND ( LastStartDate < @valueDate
              AND ( valueDate < LendDate
                    OR LendDate IS NULL
                  )
            )
)

```

```

        )
        AND #bundle1.idItemType = 17 --CombinedFacility
--SELECT * FROM #bundle1
-- Remove Orphan multi-facilities
DELETE #bundle1
WHERE
    idBundle IN (
        -- Get those idBundles of CombinedFacility which are only
        -- available in first set (i.e. only in first set(idBund
        SELECT idBundle
        FROM dbo.Bundle
        WHERE idItemtype = 17 --CombinedFacility
        EXCEPT
        SELECT idBundleUp
        FROM dbo.Bundle
        AND #bundle1.idItemType = 17
    )
--step 7 : remove customer that don't have facilities and multifaci
--Customers that don't have any facility
DELETE FROM #bundle1
WHERE
    idBundle IN ( SELECT idBundle
        FROM dbo.Bundle
        WHERE idItemType = 1
        EXCEPT
        SELECT idBundleUp
        FROM dbo.Bundle
        WHERE idItemType = 2 )
        AND #bundle1.idItemType = 1 --Customer
    END
END
BEGIN
    INSERT INTO #bundle1
    ( idBundle ,
      idBundleUp ,
      lft ,
      rgt ,
      idTree ,
      valueDate ,
      asOfDate ,
      type ,
      idLocalBL ,
      localBL ,
      idFrontOfficeCode ,
      frontOfficeCode ,
      idCollateralStructure
    )
    SELECT DISTINCT
        B.idBundle ,
        B.idBundleUp ,
        B.lft ,
        B.rgt ,
        B.idTree ,
        B.valueDate ,
        B.asOfDate ,
        B.type ,
        F.idLocalBL ,
        BL.localBL ,
        F.idFrontOfficeCode ,
        F.frontOfficeCode ,
        FC.idCollateralStructure
    FROM
        #bundle AS B
        INNER JOIN Facility F ON B.idBundle = F.idBus
        INNER JOIN Ref_LocalBL As BL ON F.idLocalBL =
        INNER JOIN Limit L ON F.idBundle = L.idBus
        AND ( L.startDate <= @valueDate
            AND ( @valueDate
                OR L.endDate
            )
        )
        INNER JOIN FacilityComp FC ON F.idFacility
        INNER JOIN Ref_FrontOfficeCode FD ON FC.idFO
        INNER JOIN #tbcodesTable tbcable ON fdTable.t
    WHERE
        ( IsActive = 1
            AND ( F.idFacilityState = 1
                OR F.idFacilityState = 0
                AND F.stateDate > @valueDate
            )
        )
        OR ( IsActive = 0
            AND ( F.idFacilityState = 0
                AND F.stateDate <= @valueDate
            )
        )
        AND ( F.idLocalBL = #localizationSG
            OR #localizationSG IS NULL
        )
        AND ( FC.idFOCode = #idFOCode
            OR #idFOCode IS NULL
        )
        AND ( FC.isMigrated = #isMigrated
            OR #isMigrated IS NULL
        )
        AND ( FC.idNCRStatus = #idNCRStatus
            OR #idNCRStatus IS NULL
        )
        AND ( FC.idCollateralStructure = #idCollatera
            OR #idCollateralStructure IS NULL
        )
    )
    INSERT INTO #bundle1
    ( idBundle ,
      idBundleUp ,
      lft ,
      rgt ,
      idTree ,
      valueDate ,
      asOfDate ,
      type ,
      type
    )
    SELECT DISTINCT
        B.idBundle ,
        B.idBundleUp ,
        B.lft ,
        B.rgt ,
        B.idTree ,
        B.valueDate ,
        B.asOfDate ,
        B.type ,
        C.type AS 'Facility'
    FROM
        #bundle AS B
        INNER JOIN #bundle1 arc ON B.idBundle = arc.idBus
        INNER JOIN #bbMultiFacilityHeader m ON arc.idB
        INNER JOIN dbo.BBMultiFacilityDetail HFD ON m
        AND sta
    WHERE
        arc.type = 'Facility'

--SUINIT - 16-June-2011 Manually merged from iTrack_August2011Release
    INSERT INTO #bundle1
    ( idBundle ,
      idBundleUp ,
      lft ,
      rgt ,
      idTree ,
      valueDate ,
      asOfDate ,
      type ,
      type
    )
    SELECT DISTINCT
        B.idBundle ,
        B.idBundleUp ,
        B.lft ,
        B.rgt ,
        B.idTree ,
        B.valueDate ,
        B.asOfDate ,
        B.type ,
        C.type AS 'Customer'
    FROM
        #bundle AS B
        INNER JOIN #bundle1 arc ON B.idBundle = arc.idBus
        INNER JOIN customer C ON arc.idBundleUp = c.i
    WHERE
        arc.type = 'Facility'
        OR arc.type = 'CombinedFacility'

    INSERT INTO #bundle1
    ( idBundle ,
      idBundleUp ,
      lft ,
      rgt ,
      idTree ,
      valueDate ,
      asOfDate ,
      type ,
      type
    )

```

```

IF @@ROWCOUNT = 0
PRINT 'Warning: prv_BundleFilter.set : no FACILITY updated'
-- ****
-- Box
-- ****

UPDATE B
SET hasMarginCallThreshold = FC.hasMarginCallThreshold ,
thresholdType = FC.thresholdType ,
thresholdValue = FC.thresholdValue ,
idMarginCallAlertScope = FC.idMarginCallAlertScope ,
idMarginAlertScope = FC.idMarginAlertScope ,
idLocalBL = idLocalBL ,
idFrontOfficeCode = FC.idFrontOfficeCode
FROM #bundle1 B
INNER JOIN Facility F ON B.idBundleUp = F.idBundle
LEFT OUTER JOIN FacilityComp FC ON F.idFacility = FC.idFacility
WHERE ( F.idFacilityState = 1
OR ( F.idFacilityState = 0
AND F.stateDate > @valueDate
)
)
UPDATE B
SET type = 'Box' ,
idItemType = 1 -- Ref ItemType
FROM #bundle1 B
WHERE idKey = CONVERT(VARCHAR(50), Box.idBundle) ,
label = Box.label ,
callcenter = L.calixdev
#bundle1 AS B
INNER JOIN Limit AS L ON B.idBundleUp = L.idBundle
INNER JOIN Box ON Box.idBundle = B.idBundle
AND ( Box.startDate < @valueDate
AND ( Box.endDate > @valueDate
OR Box.endDate IS NULL
)
)
WHERE ( L.startDate < @valueDate
AND ( @valueDate < L.endDate
OR L.endDate IS NULL
)
)
UPDATE B
SET LocalBL = RLBL.label
FROM #bundle1 B
INNER JOIN ref_localBL RLBL ON RLBL.idLocalBL = B.idLocalBL

UPDATE B
SET FrontOfficeCode = RFDC.label
FROM #bundle1 B
INNER JOIN ref_frontofficecode RFDC ON RFDC.idFrontOfficeCode

-- si le type de valo est nul alors on r?cup?re le type par d?faut
IF @@ROWCOUNT = 0
PRINT 'Warning: prv_BundleFilter.set : no BOX updated'

--MINNER --
-- Set the sharing
UPDATE B
SET minNCR = CASE WHEN B.idBundle = F.idBundle
AND FC.idMarginCallAlertScope IN ( 2, 3 )
THEN CAST(FC.coverageRatioPercent AS FLOAT) / 1
WHEN B.idBundle < F.idBundle
AND FC.idMarginCallAlertScope IN ( 1, 3 )
THEN CAST(FC.coverageRatioPercent AS FLOAT) / 1
ELSE 0.0
END ,
imperfectlySecuredLimit = CASE WHEN B.idBundle = F.idBundle
THEN ( SELECT TOP 1
ISNULL(dbo.Li
)
FROM dbo.Limit
INNER JOIN Fa
WHERE dbo.Limit.end
AND Fa.end
)
-- pour facilite s' t'a
-- Modified by Nishant for Itrack :- 462298
ELSE 0.0
END
FROM #bundle1 B
INNER JOIN Facility F ON B.idBundle = F.idBundle
INNER JOIN FacilityComp FC ON FC.idFacility = F.idFacility
--Optimisation#
INSERT #bundle
SELECT *
FROM #bundle
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_NULLS ON
GO
GRANT EXECUTE ON [dbo].[prv_BundleFilter.set.old] TO [tct_execsp]
GO

```

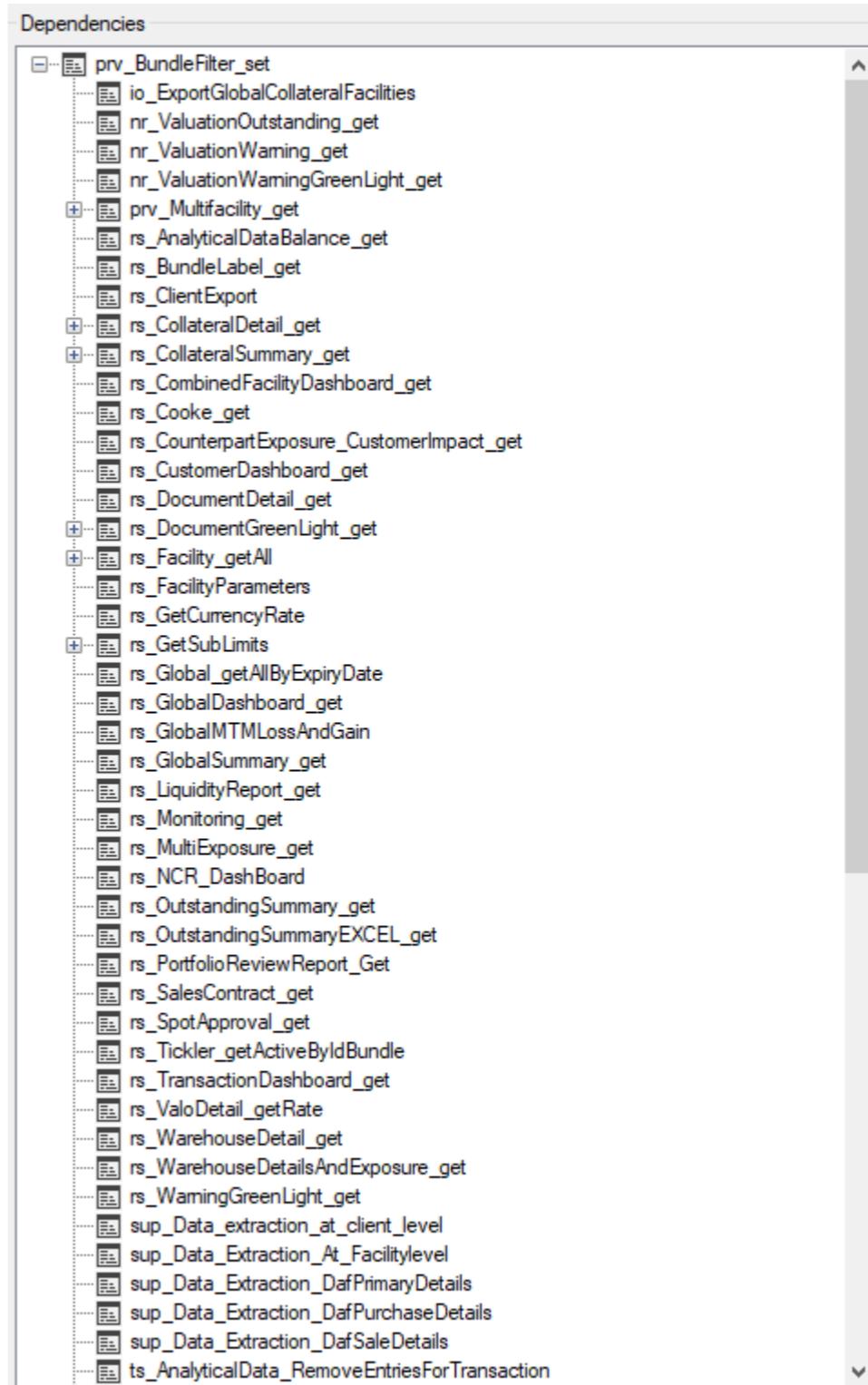
**“DON’T PANIC
I KNOW LITTLE REFACTORING
LET’S DO IT!!”**

102 COLUMNS

```
CREATE PROCEDURE dbo.prv_BundleFilter_set_old
    @valueDate DATETIME ,
    @asOfDate DATETIME ,
    @idBundle INT = NULL ,
    @localizationSG INT = NULL ,
    @idFOCode INT = NULL ,
    @isMigrated BIT = NULL ,
    @idNCRStatus INT = NULL ,
    @idCollateralStructure INT = NULL ,
    @isActive INT = 1 ,
    @idProfile INT = NULL
AS
    DECLARE @foCodesTable TABLE
    (
        idFrontOfficeCode INT
    )
    INSERT INTO @foCodesTable
    SELECT *
    FROM fn_GetDistinctFOCodes(@idProfile)
    DECLARE @CombinedFacility TABLE
    (
        idBundle INT NULL INDEX IX_idbundle,
        idBBMultiFacilityHeader INT NULL ,
        isActive BIT
    )
```

```
CREATE FUNCTION dbo.ts_Bundle_create
(
    @valueDate datetime,
    @asOfDate datetime,
    @idBundle int null INDEX idx_idbundle
    ,@idCollateralType int null
    ,@idCollateralHeader int null
    ,@idTree int null
    ,@idOutstandingValo int null
    ,@idOutstandingValoCOO int null
    ,@idValo int null
    ,@valueDate datetime null
    ,@asOfDate datetime null
    ,@itemType int null
    ,@type varchar(50) null
    ,@idKey varchar(50) null
    ,@label varchar(255) null
    ,@calInorder char(1) null
    ,@calInorderV100 char(3) null
    ,@idCollateralStructure int null
    ,@localID varchar(255) null
    ,@idFrontOfficeCode int null
    ,@FrontOfficeCode varchar(255) null
    ,@isMigrated bit null
    ,@idNCRStatus int null
    -- Global Summary Information
    ,@limitAmount float null
    ,@limitAmountCOO float null
    ,@collateralAmount float null
    ,@grossCollateralAmount float null
    ,@outstandingCOOAmount float null
    ,@UNDNE by AF : valo outstanding(to be removed) ,@outstandingIntradayAmount float nu
    ,@outstandingTheoreticalAmount float null -- relatief aux BB
    ,@facilityAvailableCOOAmount float null -- (limit - outstanding)
    ,@multifacilityAvailableCOOAmount float null -- (limit - outstanding)
    ,@collatPositionCOOAmount float null -- (collat - outstanding)
    ,@availableForDrawingCOOAmount float null -- min(facilityAvailableAmount, availableFor
    ,@facilityAvailableCurrentAmount float null -- (limit - outstanding)
    ,@multifacilityAvailableCurrentAmount float null -- (limit - outstanding)
    ,@collatPositionCurrentAmount float null -- (collat - outstanding)
    ,@availableForDrawingCurrentAmount float null -- min(facilityAvailableAmount, available
    ,@SGSummaryInformation
    ,@SGLimitAmount float null
    ,@SGLimitAmountCOO float null
    ,@SGCollateralAmount float null
    ,@SGCurrentCollateralAmount float null -- just for shared BB current collat
    ,@SGCurrentCrossCollateralAmount float null -- just for shared BB current collat
    ,@SGOutstandingCOOAmount float null
    -- UNDNE by AF : valo outstanding(to be removed) ,@SGOutstandingIntradayAmount float
    ,@SGOutstandingCurrentAmount float null
    ,@SGOutstandingTheoreticalAmount float null
    ,@SGFacilityAvailableCOOAmount float null -- (limit - outstanding)
    ,@SGMultifacilityAvailableCOOAmount float null -- (limit - outstanding)
    ,@SGCollatPositionCOOAmount float null -- (collat - outstanding)
    ,@SGAvailableForDrawingCOOAmount float null -- min(facilityAvailableAmount, available
    ,@SGFacilityAvailableCurrentAmount float null -- (limit - outstanding)
    ,@SGMultifacilityAvailableCurrentAmount float null -- (limit - outstanding)
    ,@SGCollatPositionCurrentAmount float null -- (collat - outstanding)
    ,@SGAvailableForDrawingCurrentAmount float null -- min(facilityAvailableAmount, availa
    ,@minNCR float null
    ,@imperfectlySecuredLimit float null
    ,@imperfectlySecuredAmount float null
    ,@imperfectlySecuredAvailable float null
    ,@unsecuredAmount float null
    ,@perfectlySecuredAmountCOO float null
    ,@imperfectlySecuredAvailableCOO float null
    ,@unsecuredAmountCOO float null
    ,@SGPerfectlySecuredAmount float null
    ,@SGImperfectlySecuredAvailable float null
    ,@SGUnsecuredAmount float null
    ,@MarginCallAmount float null
    ,@hasMarginCallThreshold int null
    ,@thresholdType char(1) null
    ,@thresholdValue float null
    ,@idMarginCallAlertScope int null
    ,@idNCRAlertScope int null
    ,@grossODRatio float null
    ,@netODRatio float null
    ,@grossCurrentRatio float null
    ,@netCurrentRatio float null
    ,@idCollateralStructure int null
    ,@idCollateralHeader int null
    ,@idLimitSGAmount int null
    ,@sharingSG float null
    ,@sharingSGCOO float null
    ,@sharingSGCurrent float null
    ,@sharedBB int null
    ,@idNCRapart int null
    ,@idNCRHeader int null
    ,@Cookie float null
    ,@SyndicatedCookie float null
    ,@OutstandingCookie float null
    ,@UnusedPortionCookie float null
    ,@committed int null
    ,@unaggregated datetime null
    ,@expiryDate datetime
    ,@maturityDate datetime
    ,@effectiveDate DATETIME
    -- Added by Anant for ittrack 128417B
    ,@initialOutstandingAmount float null
    ,@initialDataForGlobalHeader float null
    ,@grossInclSGCollateralRatio FLOAT NULL
    ,@grossInclSGCollateralRatioFLOAT FLOAT NULL
    ,@grossInclSGCollateralAmount FLOAT NULL
    ,@grossInclSGGlobalCollateralAmount FLOAT NULL
    ) AS
BEGIN
    RETURN
END
```

100+ DEPENDENCIES



I WAS STILL
SCRATCHING
THE SURFACE



**FOLLOWING
THE EXISTING
PATTERN**

**TIME
PRESSURE**

**LACK OF
REGULATIONS**

**UNFAMILIAR
CODE**

**NEW TO
TEAM**

**DON'T KNOW
WHEN IT
HAPPENED**

**INEXPERIENCED
DEVELOPERS**

MAKING
SMALLEST
POSSIBLE CHANGE

HARD
TO
UNDERSTAND

**WHY
THIS STORED
PROCEDURE
BECAME LIKE THIS?**

ATTEMPT 1

SMALL CHANGE & IT FAILED

CODE IS RIGID

ATTEMPT 20+

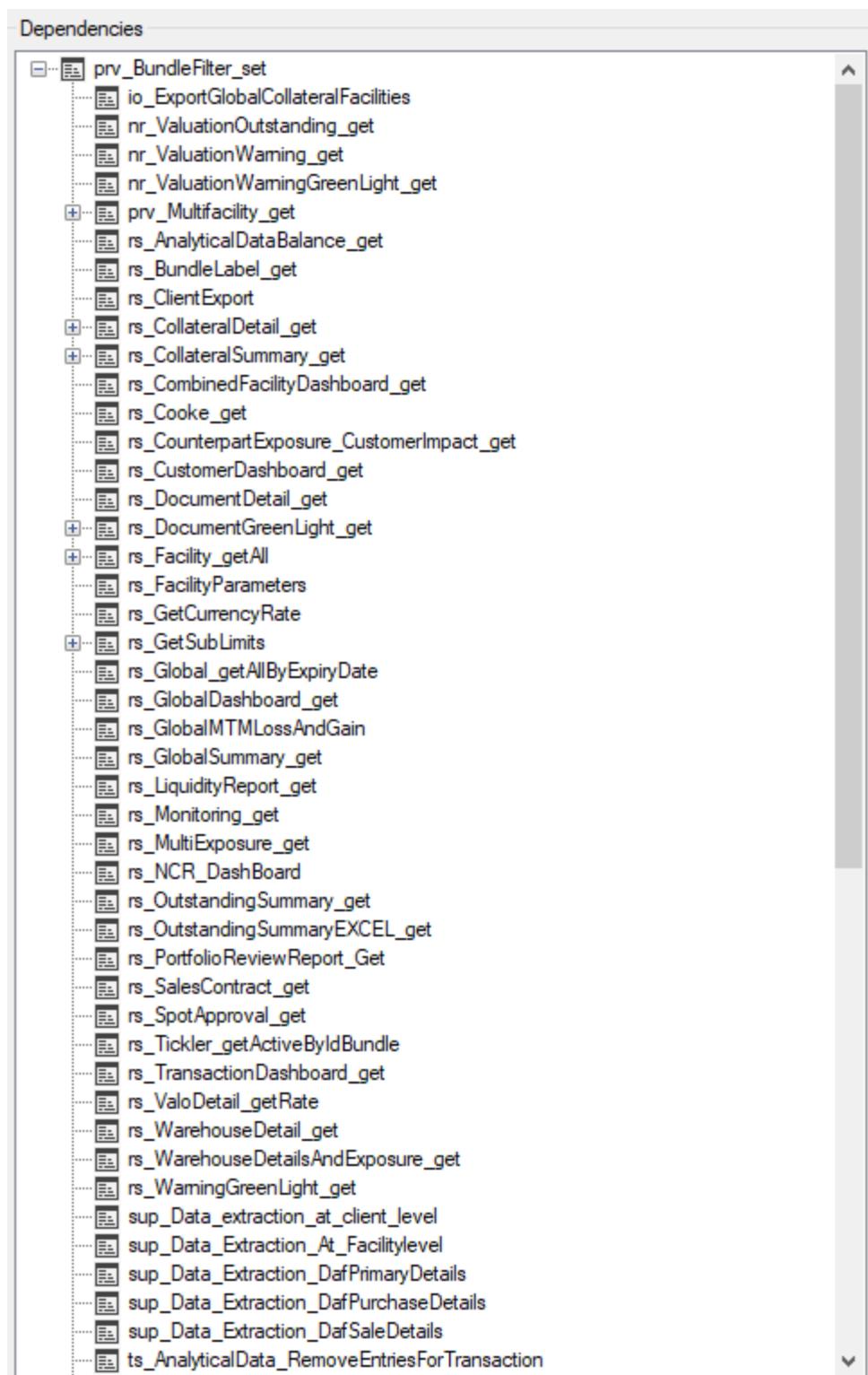
Made changes, it worked!!!
Checked in.

WE RELEASED ☺

After **2 Days** in production
It broke the reports
from completely different part of
application

CODE IS FRAGILE

100+ DEPENDENCIES



When looked deeper into
the code, dependencies
have grown because this
module
**couldn't be reused
partly**

CODE IS IMMOBILE

Module was full of
technical names than
business/domain names

CODE IS NOT READABLE

**PERFECT
INGREDIENTS
FOR?**



FEAR





TESTS
FEAR

TEST AUTOMATION

UNIT
TESTING

FUNCTIONAL
TESTING

ZJSONPATCH

DIFFY

DATABASE
NONREG PACK

Finds potential bugs in your system
comparing the output of
new code and your ***old code***.

STORED PROCEDURE OUTPUT COMPARER

STAND ALONE

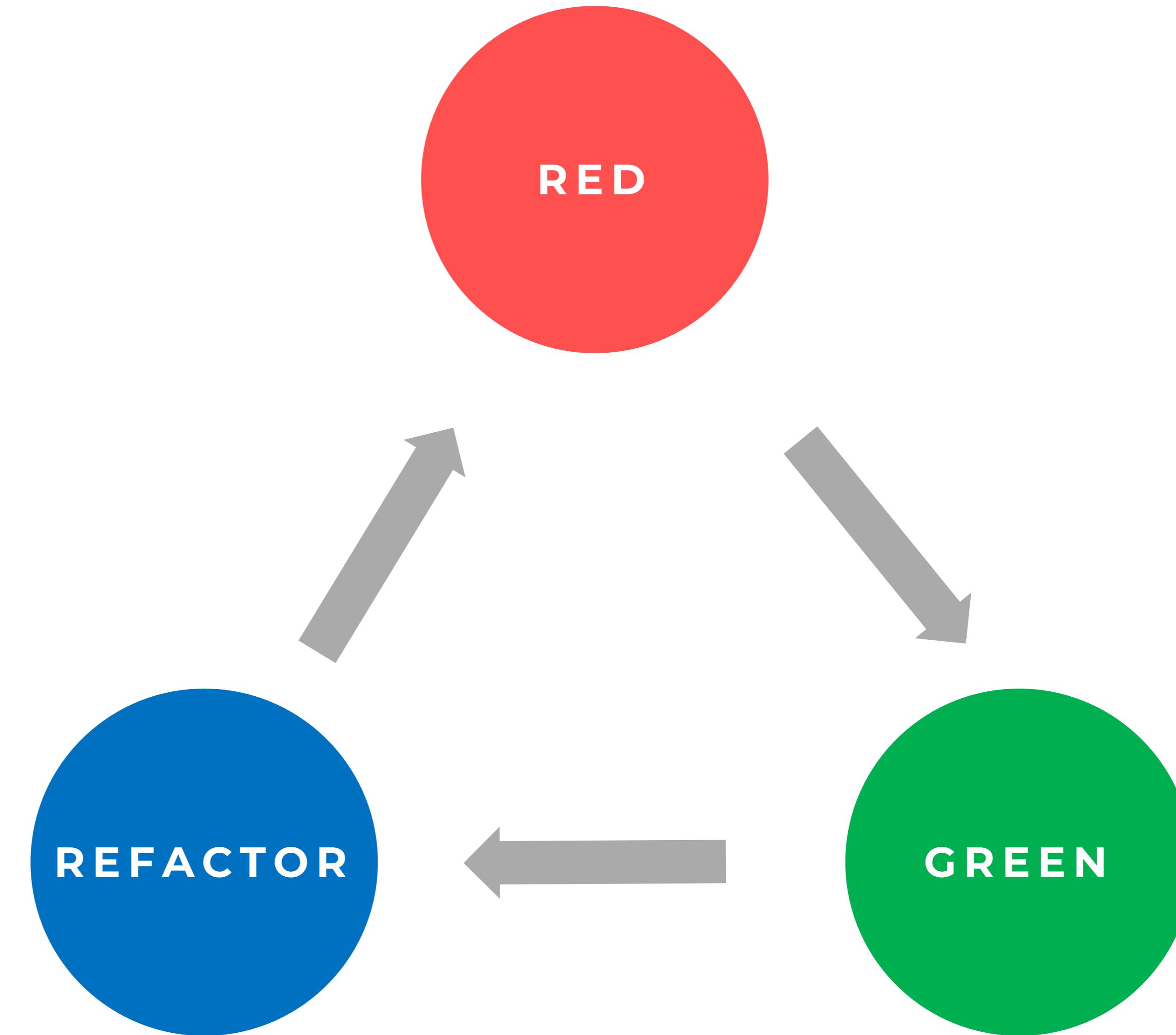
The screenshot shows the SPComparer application window. On the left, there's a 'Compare From' section with dropdowns for 'Server - Database' (SRVCLDTCTI004\MSPARTCTU04 TomcatData_Perf) and 'Stored Procedure' (dbo.ts_FacilityBlockedCash_EconomicMargin). To its right is a 'Compare With' section with similar dropdowns. Below these are 'Parameters' and a list of parameter values. In the center, there's a 'Results' panel showing a summary: 'How many times to run?' (1), 'Parallel Users' (10), 'RUN' button, 'RUN Parallel' button, 'RUN VAL' button, and '0 of 0'. Below this are 'Time for New' (1616 ms) and 'Time for Old' (2488 ms) with their respective standard deviations. A checkbox for 'Oracle?' is present, and a checked checkbox for 'Break When Not Matching' is shown with an 'Avg Diff' of 1112.5. At the bottom, a red message says 'Result Data is not same for Table and for Column idTransactionVersion'. Two tables are displayed side-by-side, each with columns: idTransactionVersic, idBox, BlockedCashAmou, totalAmount, and currency. The first table has rows for 81092, 82083, 82422, 82634, and 82868. The second table has rows for 82083, 82868, 82634, 82422, and 81092. Both tables show identical data. At the very bottom, a section titled 'Failed SP with Parameters' lists 'dbo.ts_FacilityBlockedCash_EconomicMargin @bundleId = 101, @valueDate = '2019-03-22''.

INTEGRATED WITH CI/CD

The screenshot shows the Jenkins interface for a build step named 'SP_Comparer'. The top navigation bar includes 'Projects', 'Changes', 'Agents 23', 'Build Queue 11', and the user 'Bharat MANE'. The main area shows the build history for '#282 (03 Nov 20 01:00)'. The 'Performance' tab is selected. It displays two failed test cases: 'VAL.ref_CommodityQuality_get' and 'VAL.ref_Counterpart_get'. Below this is a detailed table of the build results:

Timestamp	Result	New SP Time	Old SP Time	% Diff	Command
11/3/2020 1:02:39 AM	Data is not same for Table and for Column SG_Rating	17	12	-5	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='Z' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Data is not same for Table and for Column SG_Rating	15	10	-5	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='Y' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	16	13	-3	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='X' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	19	14	-5	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='W' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Data is not same for Table and for Column SG_Rating	21	24	3	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='V' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	20	20	0	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='U' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Data is not same for Table and for Column SG_Rating	27	47	20	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='T' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	70	118	48	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='S' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	45	35	-10	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='R' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	16	12	-4	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='Q' , @activeOnly = NULL
11/3/2020 1:02:39 AM	Row count is not same for Table	31	58	27	VAL.ref_Counterpart_get @idCounterpart = NULL, @filter ='P' , @activeOnly = NULL

REFACTOR
FEARLESSLY



1

WHILE MAKING
CHANGES
TO EXISTING CODE

2

WHILE ADDING
NEW CODE



WHEN YOUR CODE WORKS THE FIRST TIME



IT'S JUST **HALF** THE JOB DONE

THE TIME THAT THE CODE FIRST
WORKS...

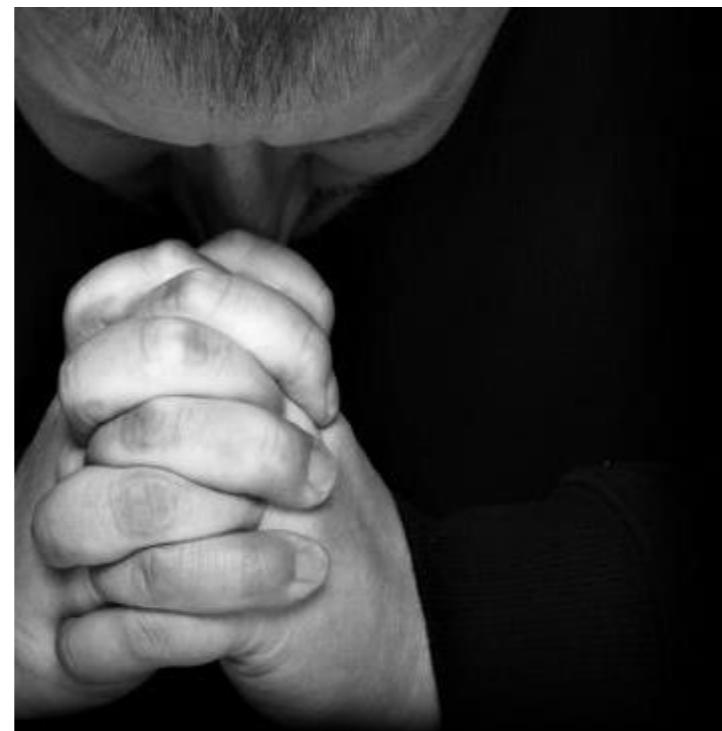
...IT'S THE MOMENT IN WHICH
YOU START **BUILDING YOUR**
CAREER

**...IT'S THE MOMENT IN WHICH
YOU START BUILDING YOUR
DESIGN SKILLS**

DOUBTS



REMORSE



**MICRO
DECISIONS**



**SELF
REVIEW**



**IT'S PERFECT MOMENT TO
CLEAN THE CODE**

NAMES	FUNCTIONS	COMMENTS
FORMATTING	OBJECTS & DATA STRUCTURES	ERROR HANDLING
CONTINUOUS REFACTORING	PAIR PROGRAMMING	MONITORING

CLEAN CODE TOPICS

NAMES

THE **POWER** INVESTED IN OUR
FINGERTIPS



local variable **private function**

instance variable

argument namespace

directory

class

global variable

public function

module

library

package

1

...BECAUSE WE NAME SO
MUCH

...WE MUST USE THIS
POWER WISELY
WE MUST DO IT WELL



TIM OTTINGER

COINED THE PHRASE,
“IMPLICITY OF A CODE”

USE INTENTION-REVEALING NAMES

```
int d; // elapsed time in days
```



```
int elapsedTimeInDays;
```

```
public List<int[]> getThem() {
    List<int[]> list1 = new ArrayList<int[]>();
    for (int[] x : theList)
        if (x[0] == 4)
            list1.add(x);
    return list1;
}
```

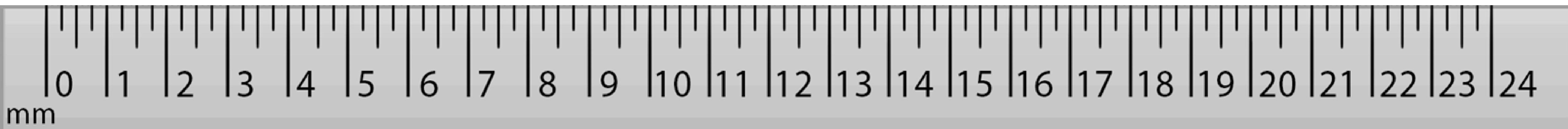


```
public List<int[]> getFlaggedCells() {
    List<int[]> flaggedCells = new ArrayList<int[]>();
    for (int[] cell : gameBoard)
        if (cell[STATUS_VALUE] == FLAGGED)
            flaggedCells.add(cell);
    return flaggedCells;
}
```

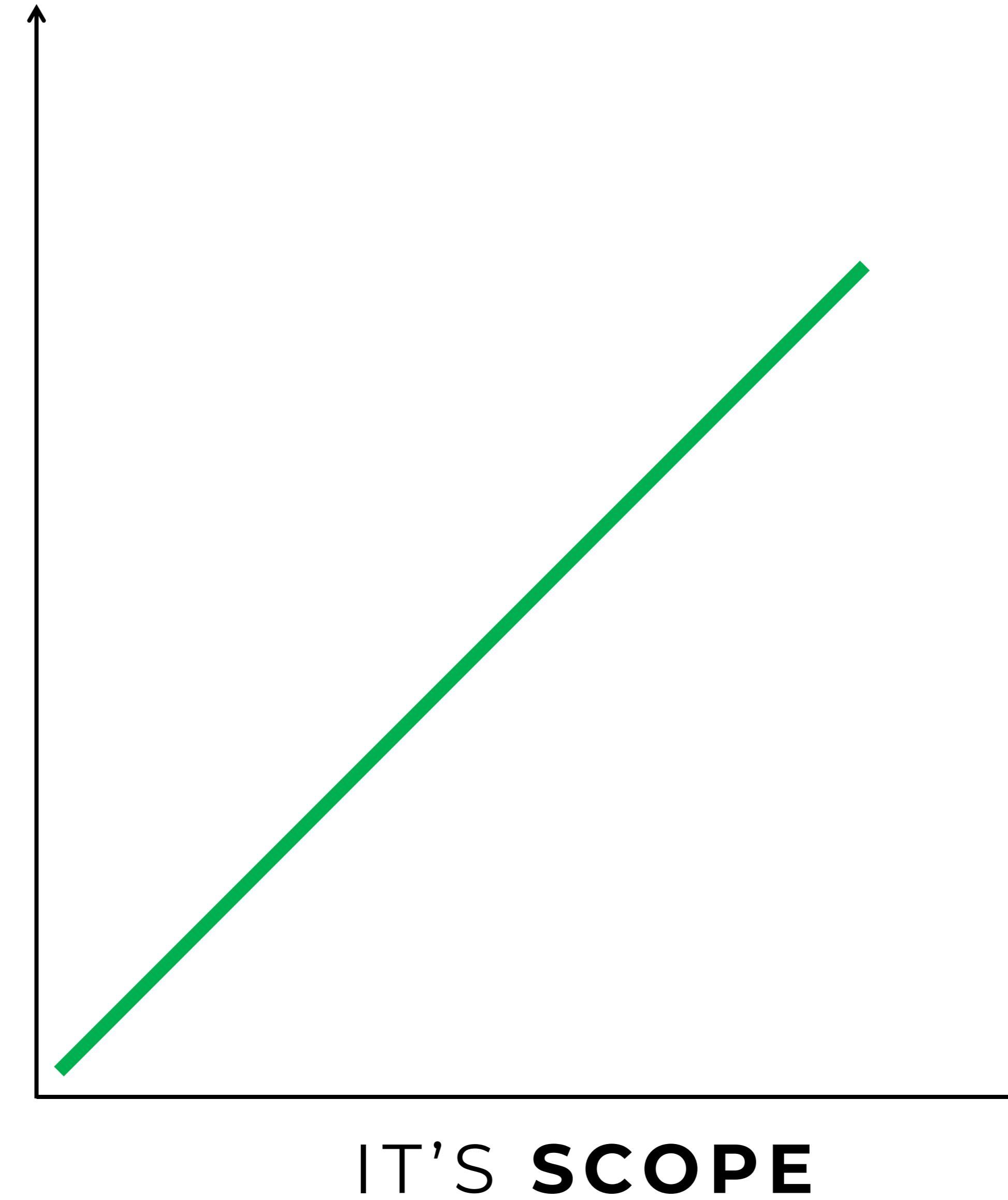
```
public List<Cell> getFlaggedCells() {
    List<Cell> flaggedCells = new ArrayList<Cell>();
    for (Cell cell : gameBoard)
        if (cell.isFlagged())
            flaggedCells.add(cell);
    return flaggedCells;
}
```

WE COMMUNICATE

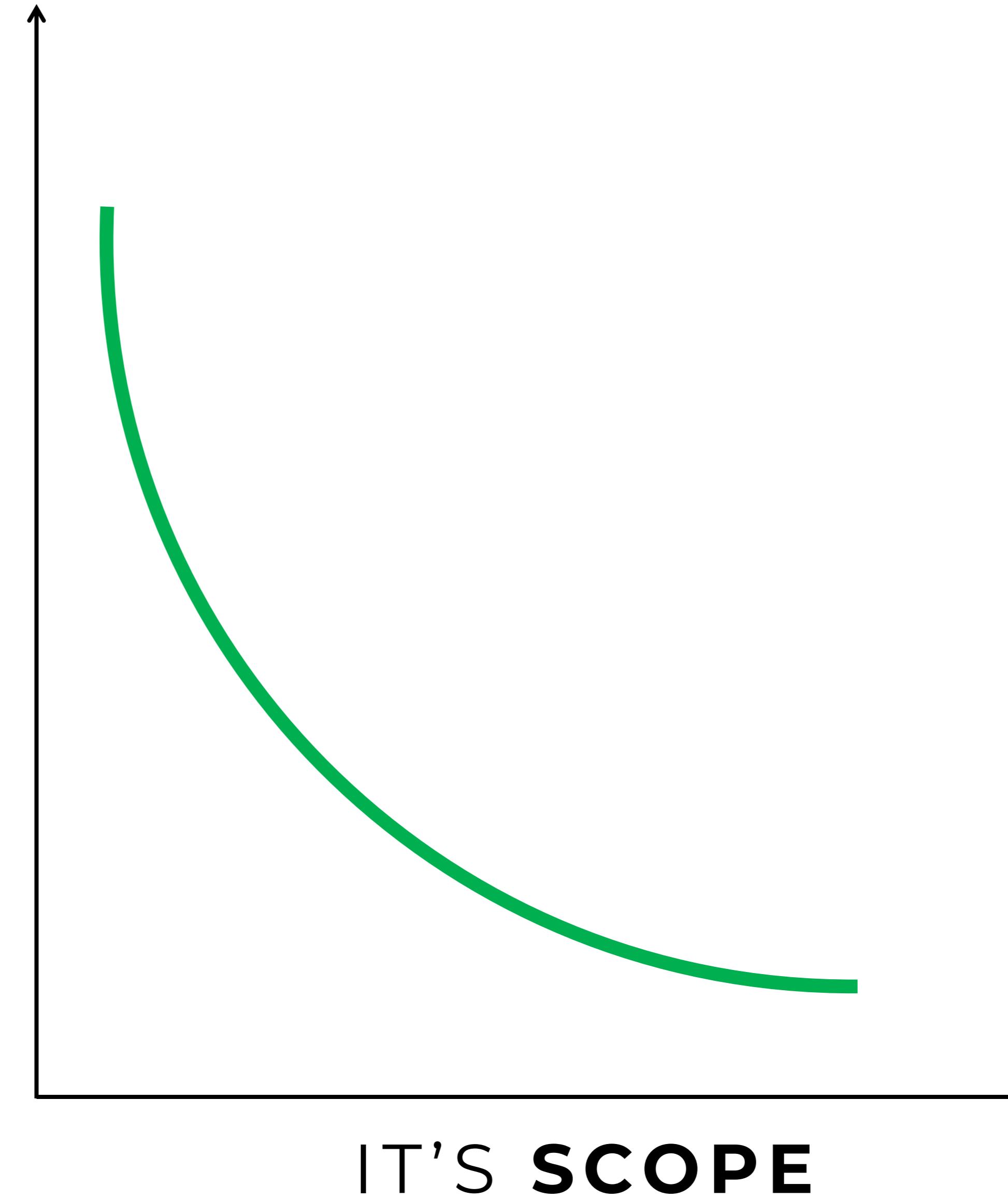
LENGTH



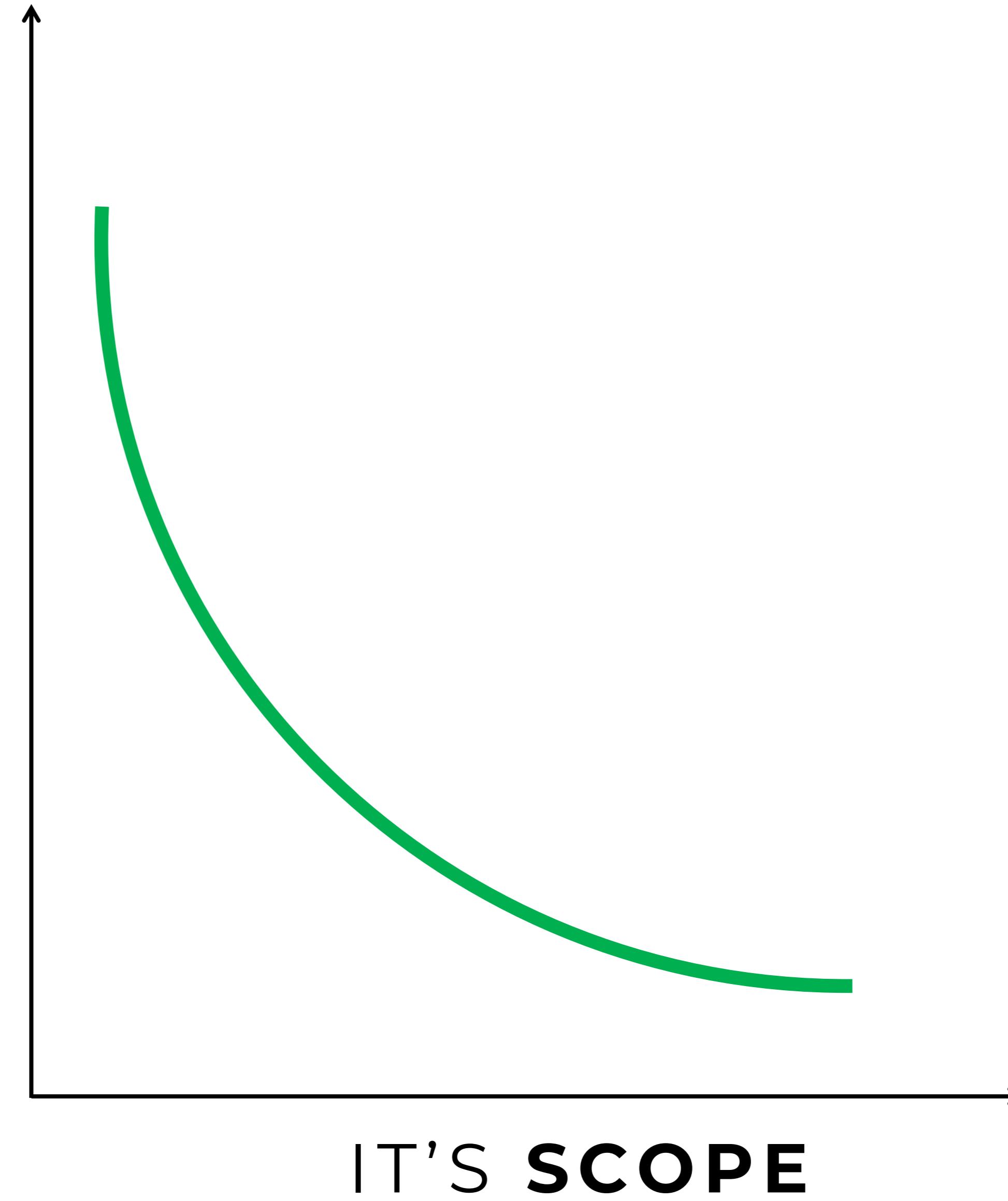
VARIABLE NAME
LENGTH



FUNCTION NAME
LENGTH



CLASS NAME
LENGTH



V 1.0

- ▶ **C# Tomcat.StoredProcedure**
- ▶ Properties
- ▶ References
- ▶ App.config
- ▶ Form1.cs
- ▶ **C# Program.cs**
- ▶ **C# Utility.cs**

V 2.0

- ▶ **C# Tomcat.SPComparer**
- ▶ Properties
- ▶ References
- ▶ Config
 - ▶ SPs
 - ▶ serverconfig.json
 - ▶ App.config
- ▶ **C# DataSetComparer.cs**
- ▶ **C# DBItem.cs**
- ▶ **C# Logger.cs**
- ▶ packages.config
- ▶ **C# Program.cs**
- ▶ SPComparer.cs
- ▶ **C# SPComparerOutput.cs**
- ▶ **C# StoredProcedureComparer.cs**
- ▶ **C# Utility.cs**

V 3.0

- ▶ ✓ **C# Tomcat.SPComparer**
- ▶ ✓ **C# Tomcat.SPComparerProvider**
- ▶ Properties
- ▶ References
- ▶ CommandOutput.cs
- ▶ + **C# ComparerConfiguration.cs**
- ▶ ✓ **C# ComparerProvider.cs**
- ▶ ComparingItem.cs
- ▶ ComparingItemCollection.cs
- ▶ DataSetComparer.cs
- ▶ DataSetComparerResult.cs
- ▶ DBItem.cs
- ▶ ExecutionOutput.cs
- ▶ ExecutionTime.cs
- ▶ + **C# IStoredProcedureExecutor.cs**
- ▶ Logger.cs
- ▶ SPComparerOutput.cs
- ▶ StoredProcedureComparer.cs
- ▶ + **C# StoredProcedureExecutor.cs**
- ▶ + **C# StoredProcedureOutputComparer.cs**
- ▶ Utility.cs

V 4.0

???

```
► C# Tomcat.StoredProcedureTester
  ► Properties
  ► References
  ▲ Interfaces
    ► C# IExecutor.cs
    ► C# ITestResultPublisher.cs
  ▲ Model
    ► C# DataSetComparerResult.cs
    ► C# MissMatchItemType.cs
    ► C# StoredProcedureInExecution.cs
    ► C# TestConfig.cs
    ► C# TestOutput.cs
    ► C# TestParameter.cs
    ► C# TestSuite.cs
    ► C# TestSuiteOutput.cs
    ► C# TestUnit.cs
    ► C# TestUnitOutput.cs
    ► C# TestUnitRunOutput.cs
  ◁ App.config
  ► C# DataSetComparer.cs
  ► C# HtmlTestResultPublisher.cs
  ► C# Loader.cs
  ► C# MSSQLExecutor.cs
  ► C# Tester.cs
  ► C# TestRunner.cs
```

THE MOMENT NAME
WAS **GIVEN** TO THIS
CLASS THE WHOLE
THING BECAME
CRYSTAL CLEAR

GOOD NAMES TELLS
THE **CONTEXT** OF
ENTIRE SYSTEM

WE COMMUNICATE

AVOID DISINFORMATION

```
10 var XYZFooBarClassForBlabla  
11 var XYZFooBarClassForBlabla  
12
```

```
4 var a = 1;  
5 if ( 0 == 1 )  
6 a = 01;  
7 else  
8 l = 01;  
9
```

```
public class Document  
public class DocumentInfo  
public class DocumentDetail  
public class Collateral  
public class Documents  
public class DocumentList
```

Using lower case **I** (looks like **number-1**) and uppercase **O** (looks like number-**0**) are also unhelpful.

“A software author must avoid leaving **false clues** which obscure the meaning of code.”
- Ottlinger

WE COMMUNICATE

USE PRONOUNCEABLE NAMES

```
// generation date, year, months, day, hour, minute, and second
class DtaRcrd102 {
    private Date genymdhms;
    /* ... */
}
```



```
// better:
class Customer {
    private Date generationTimestamp;
    /* ... */
}
```

```
@amount float,
@ca3isodev char(3),
@valueDate datetime,
@expiryDate datetime ,
```



CURRENCY

```
public void CalculateSCPUAvailableQuantity(DocumentTransfer transfer)
```

“Programming is a social activity” - Bob Martin

WE COMMUNICATE

USE SEARCHABLE NAMES

```
for (int j=0; j<34; j++) {  
    s += (t[j]*4)/5;  
}
```



```
int realDaysPerIdealDay = 4;  
const int WORK_DAYS_PER_WEEK = 5;  
int sum = 0;  
for (int j=0; j < NUMBER_OF_TASKS; j++) {  
    int realTaskDays = taskEstimate[j] * realDaysPerIdealDay;  
    int realTaskWeeks = (realTaskDays / WORK_DAYS_PER_WEEK);  
    sum += realTaskWeeks;  
}
```

```
let a = {'apple' : 2, 'mango' : 1,  
        'banana': 0, 'orange': 2,  
        'water-melon': 2}  
  
for (b in a) {  
    let c = 'no'  
    if (a[b] == 2) c = 'many'  
    else if (a[b] == 1) c = 'one'  
    else if (a[b] == 0) c = 'no'  
  
    console.log(a, 'has', c, c == 'many'? 'seeds' : 'seed')  
}
```



```
const NO_SEED = 0  
const ONE_SEED = 1;  
const MANY_SEEDS = 2;  
  
let fruits = {'apple' : MANY_SEEDS, 'mango' : ONE_SEED,  
             'banana': NO_SEED, 'orange': MANY_SEEDS,  
             'water-melon': MANY_SEEDS}  
  
// print the seeds count  
for (fruit in fruits) {  
    let count = 'no';  
  
    if (fruits[fruit] == MANY_SEEDS) count = 'many'  
    else if (fruits[fruit] == ONE_SEED) count = 'one'  
  
    console.log(fruit, 'has', count, count == 'many'? 'seeds' : 'seed')  
}
```

Replace literals with constants

CLASS NAMES

A CLASS NAME SHOULD
BE A **NOUN**,
NOT A **VERB**.

AVOID WORDS like
Manager, Processor, Data, or Info.

GOOD NAMES could be:
Tester, TestSuite, TestUnit, HtmlTestResultPublisher.

METHOD NAMES

METHODS SHOULD HAVE **VERB**

```
internal static void Consolidation(Process process,  
internal static void UnitDdaNetting(  
internal static void UnitAnalyticalOutstandingGeneration(Process process,  
public double QualityUnitConversion(int quality, int idUnit, double quantity,  
                                         int sourceDocumentIdQuality,  
                                         int sourceDocumentIdUnit,  
                                         DateTime valueDate)...
```

GOOD NAMES could be:

*Save(), Run(), Publish(), Valuate(),
Consolidate()
RunNettingValuation(),
GenerateAnalyticalOutstanding()
ConvertQualityUnit()*

WE COMMUNICATE

HOWEVER

THERE ARE NO GOOD NAMES

IT'S CONTINUOUS PROCESS
EVERY TIME YOU HAVE BETTER
UNDERSTANDING OF CODE,
YOU FIND A BETTER NAME

ENGLAND'S ETON COLLEGE



“THAT'S EASY”



JUST BRUSH
OFF THE DEW
EVERY MORNING



MOW THEM
EVERY
OTHER DAY



AND ROLL
THEM
ONCE A WEEK



DO THAT FOR **500 YEARS** &
YOU'LL HAVE A NICE LAWN, TOO

A black and white photograph of a woman with dark hair, resting her head on her hand. She is wearing a light-colored top. The background is plain and light.

CARE

tl;dr

NEGLECTION **ACCELERATES** ROT
FASTER THAN ANY OTHER FACTOR

TEST AUTOMATION GIVES YOU
POWER OF
FEARLESS REFACTORING

FIND OPPORTUNITIES OF
AUTOMATION IN ABSENCE OF
TESTS FOR OLD CODE

**USE THE GIFTED POWER OF
NAMING THINGS WISELY**

USE INTENTION-REVEALING NAMES

AVOID DISINFORMATION

USE PRONOUNCEABLE NAMES

USE SEARCHABLE NAMES

CLASS NAMES MUST BE **NOUNS**
METHOD NAMES MUST BE **VERBS**

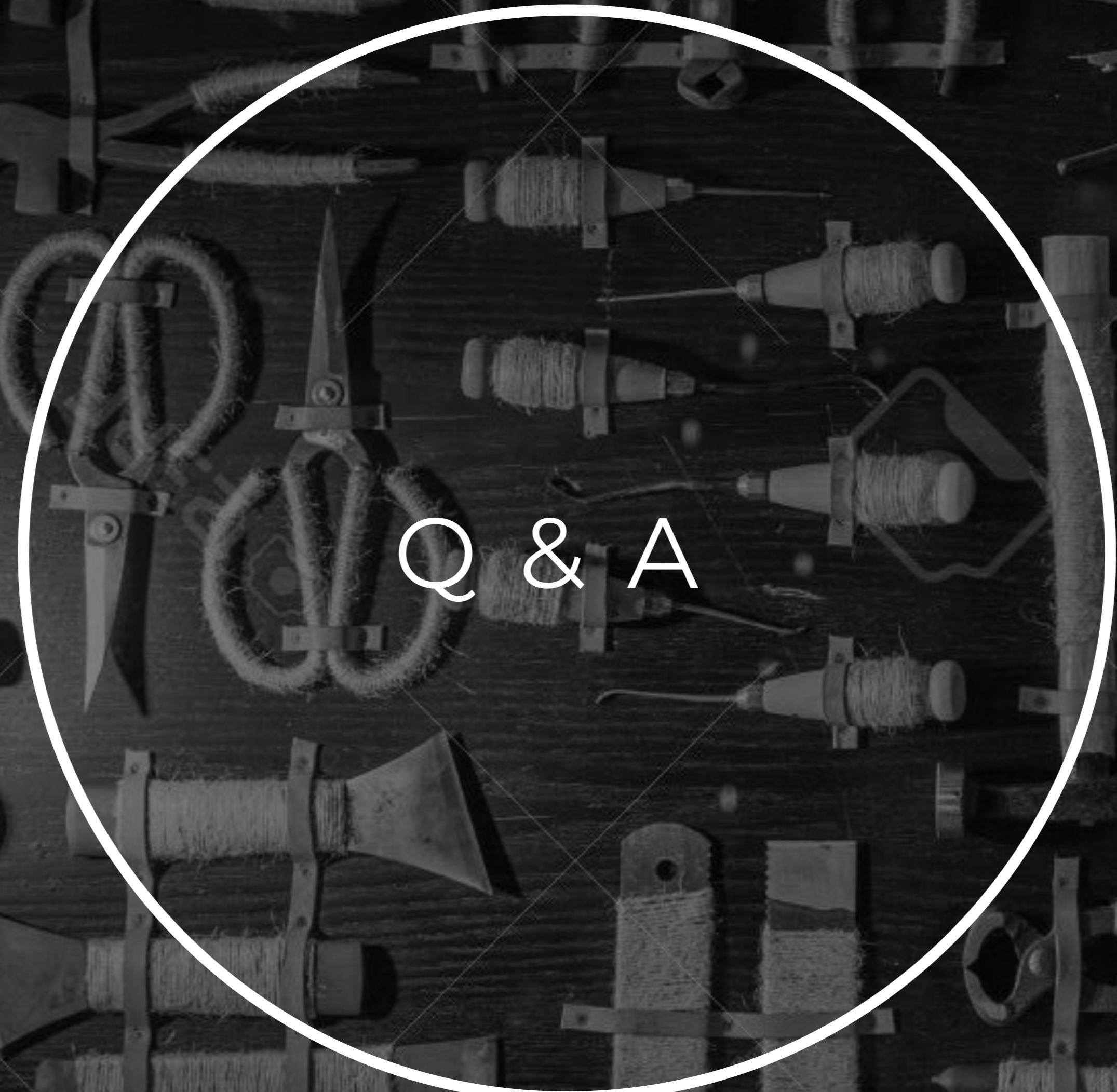
REMEMBER WE DON'T CODE,
WE COMMUNICATE

REMEMBER WE DON'T CODE,
WE COMMUNICATE

DON'T FEAR
&
TAKE CARE

MINDSET

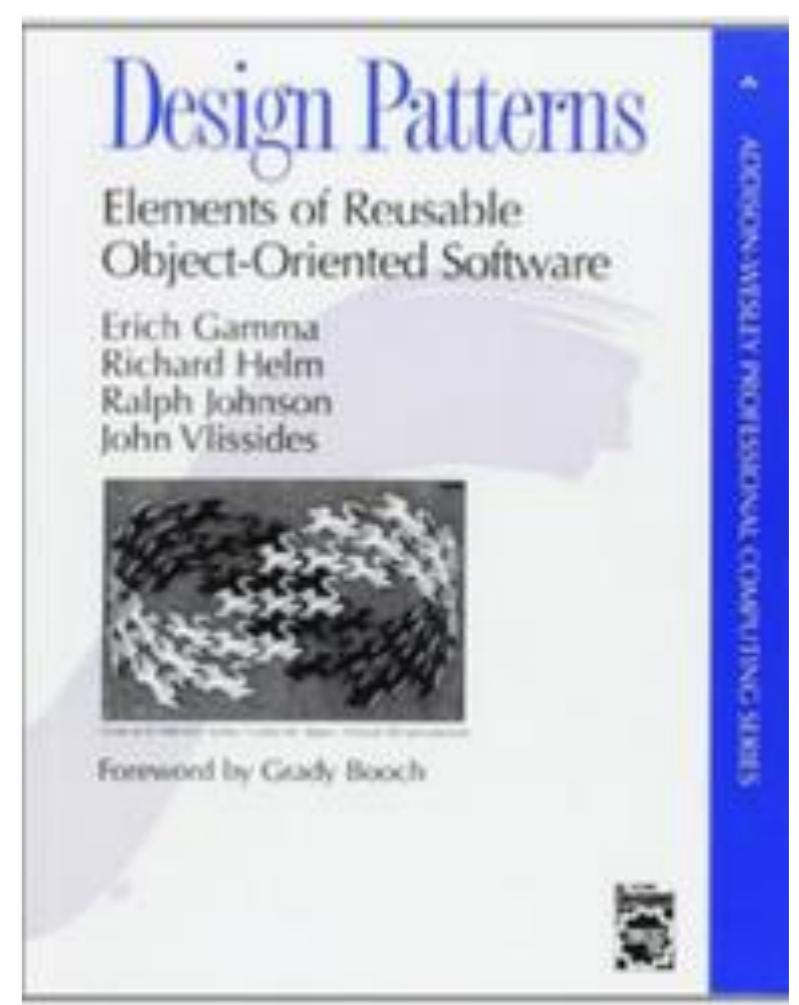
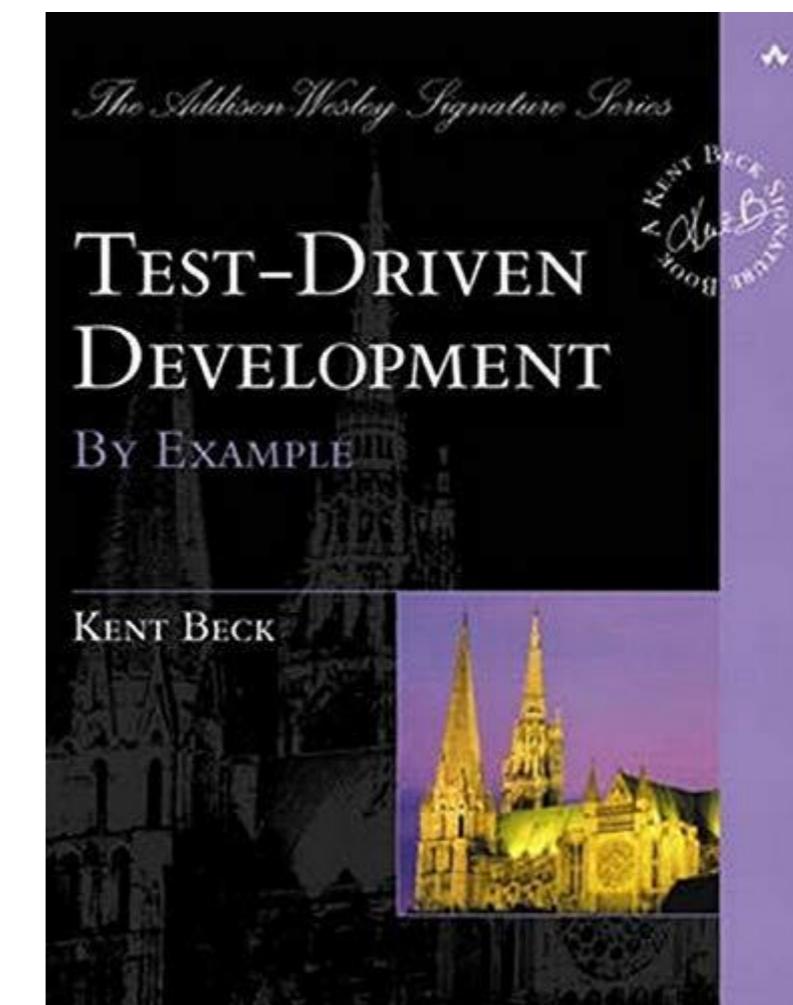
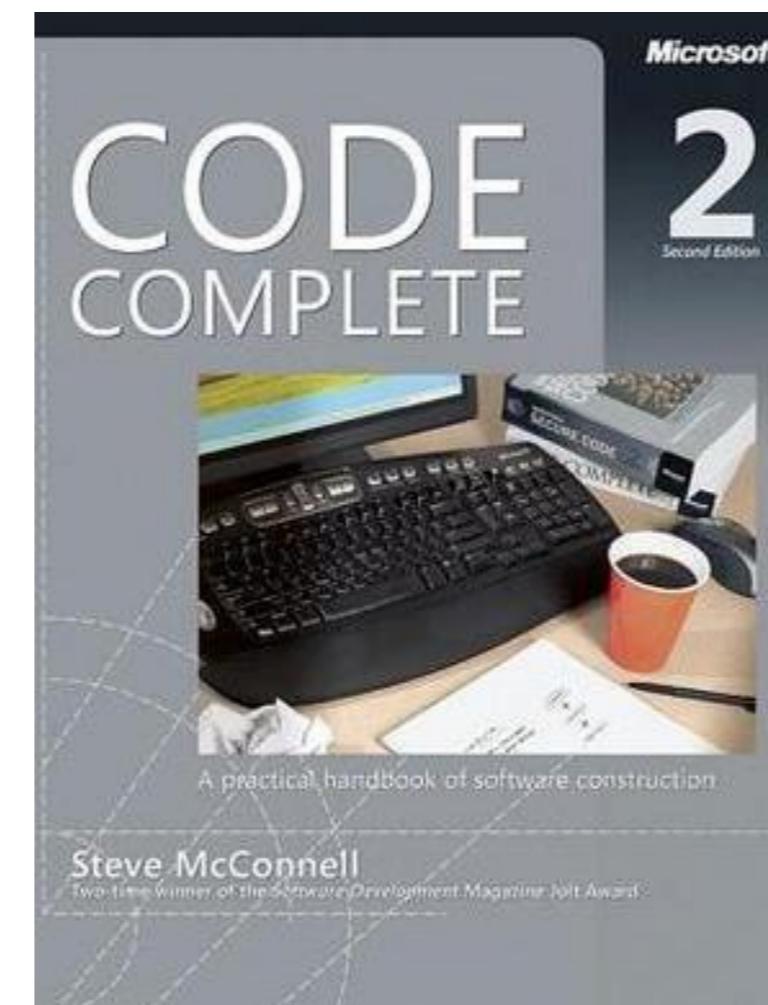
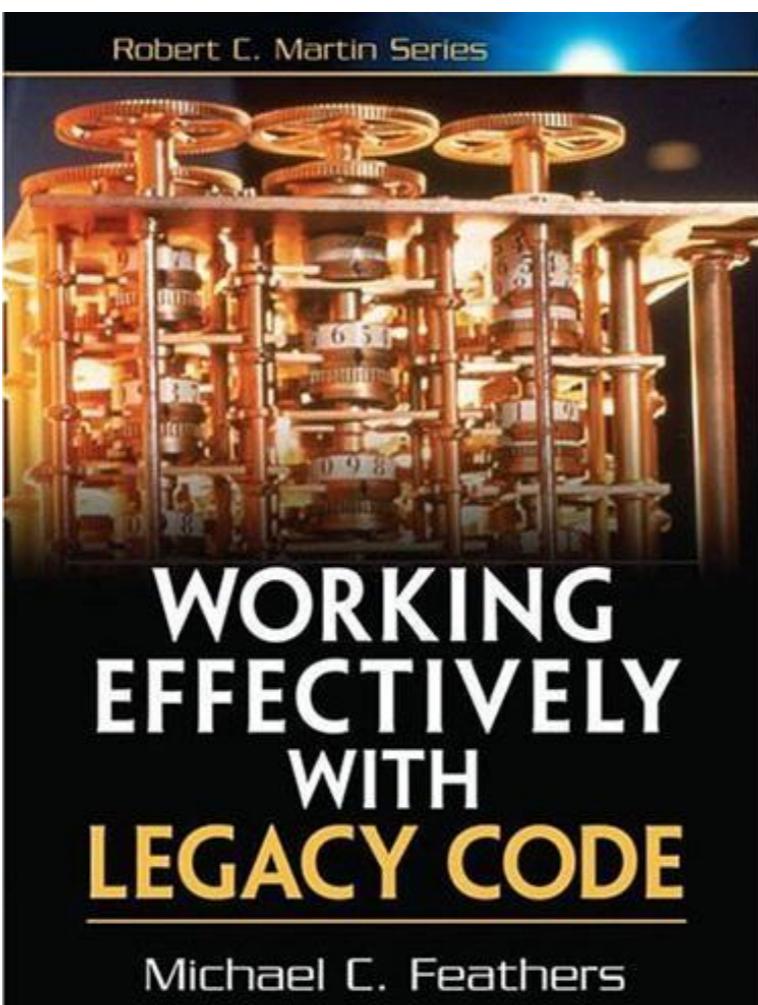
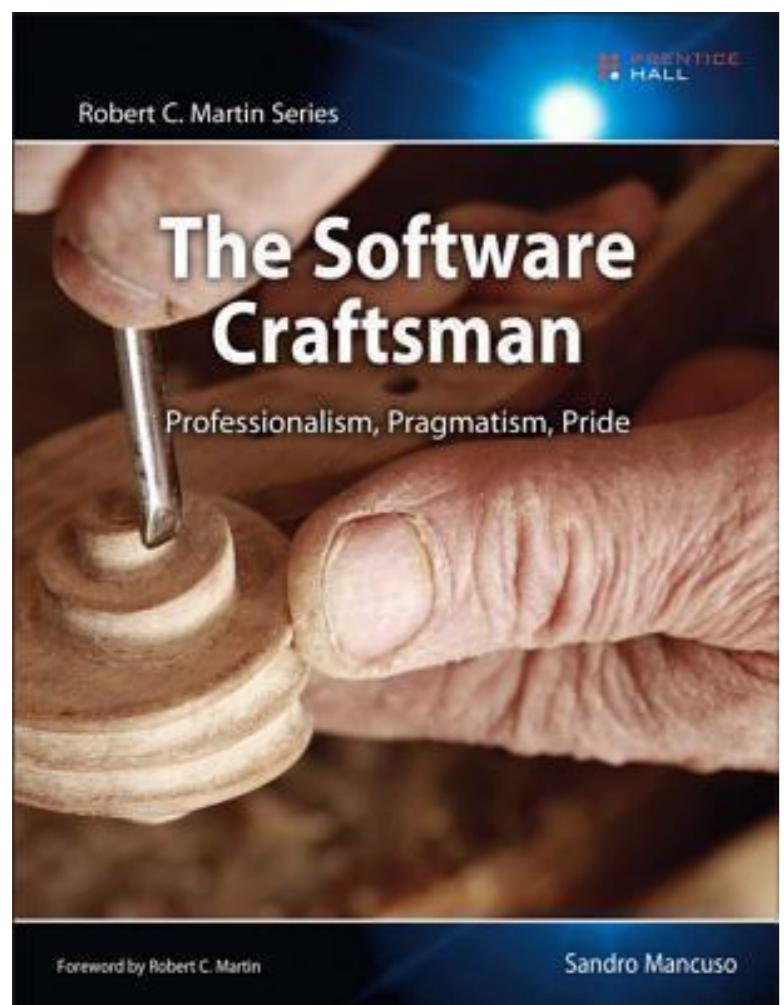
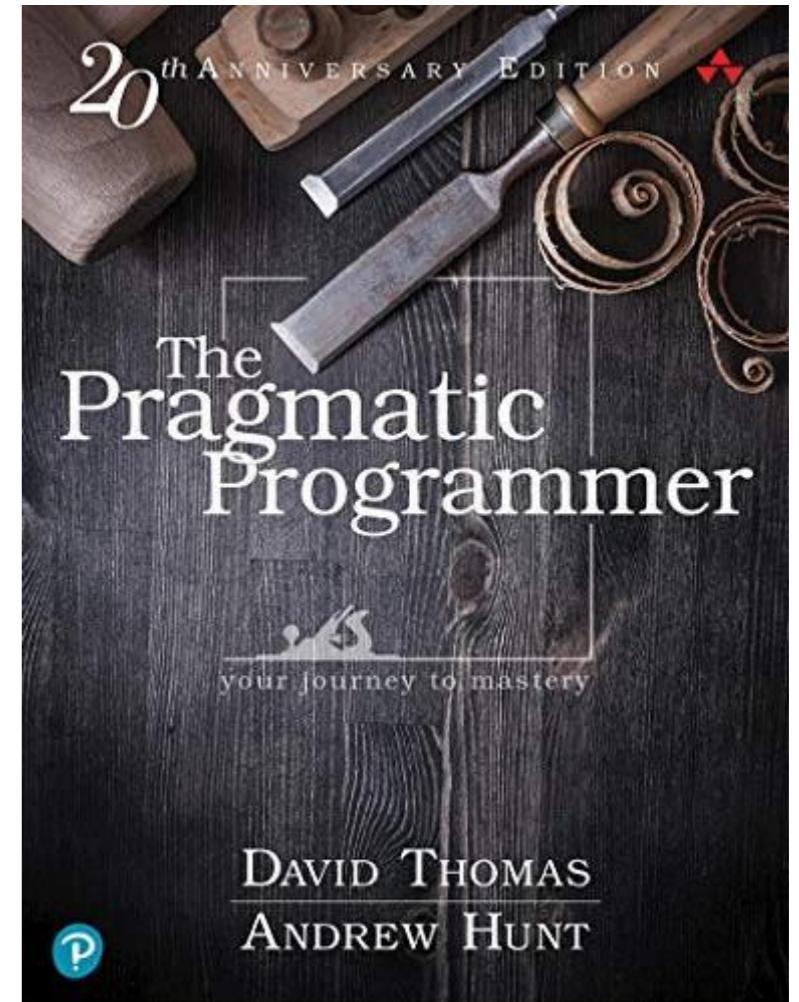
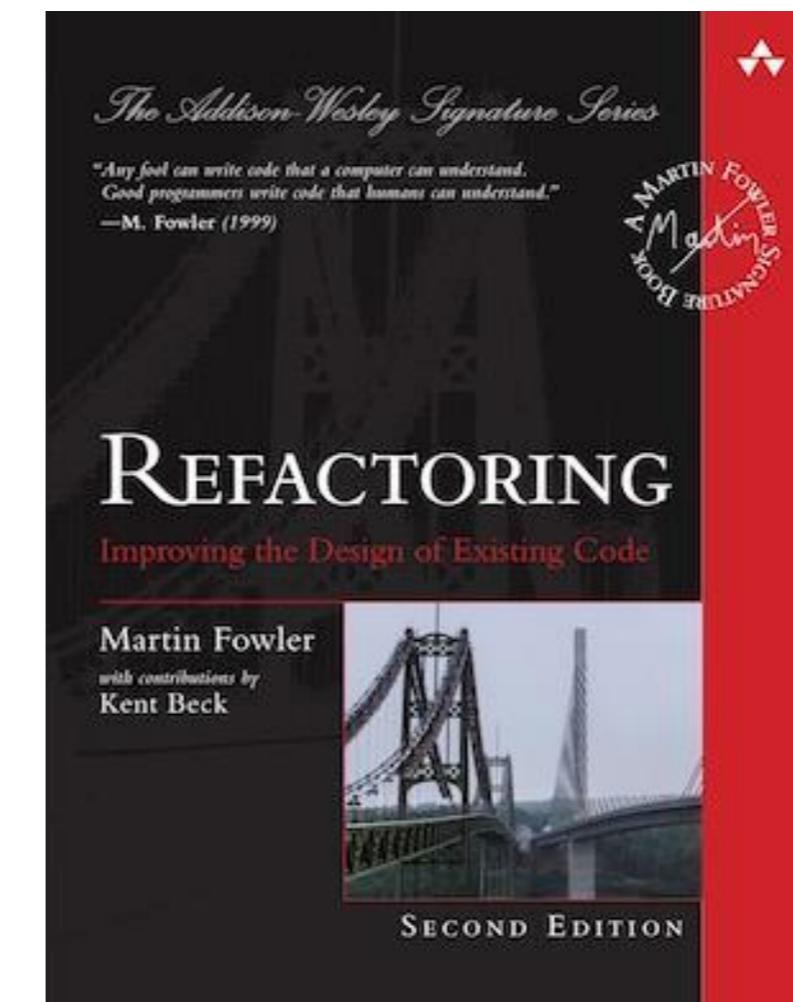
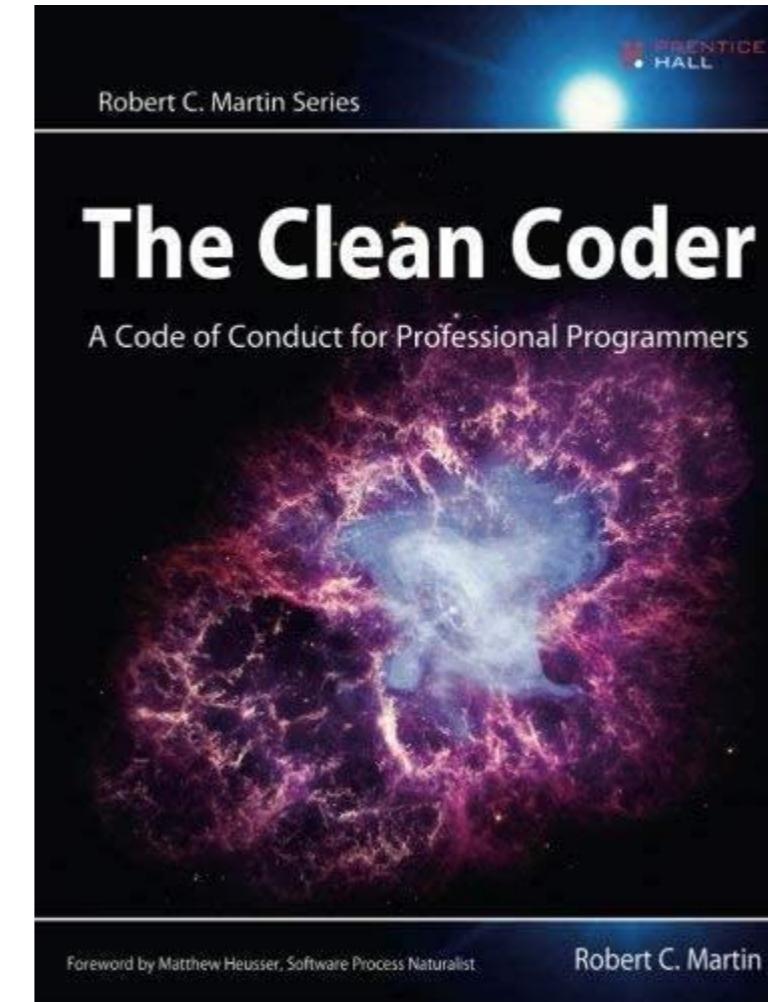
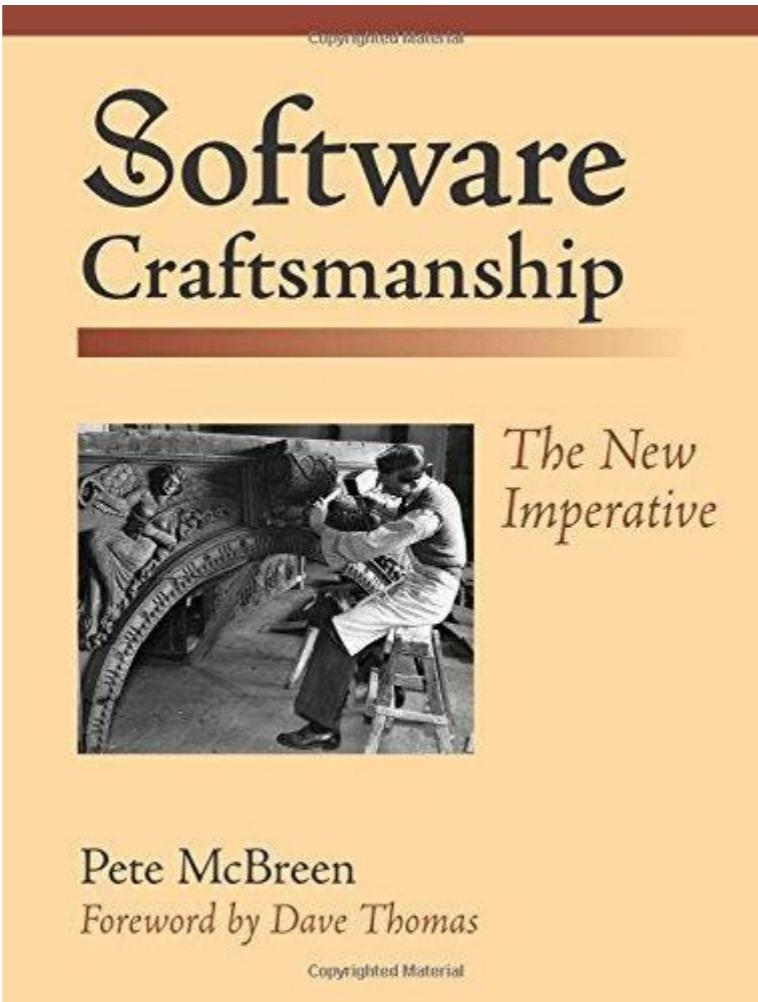
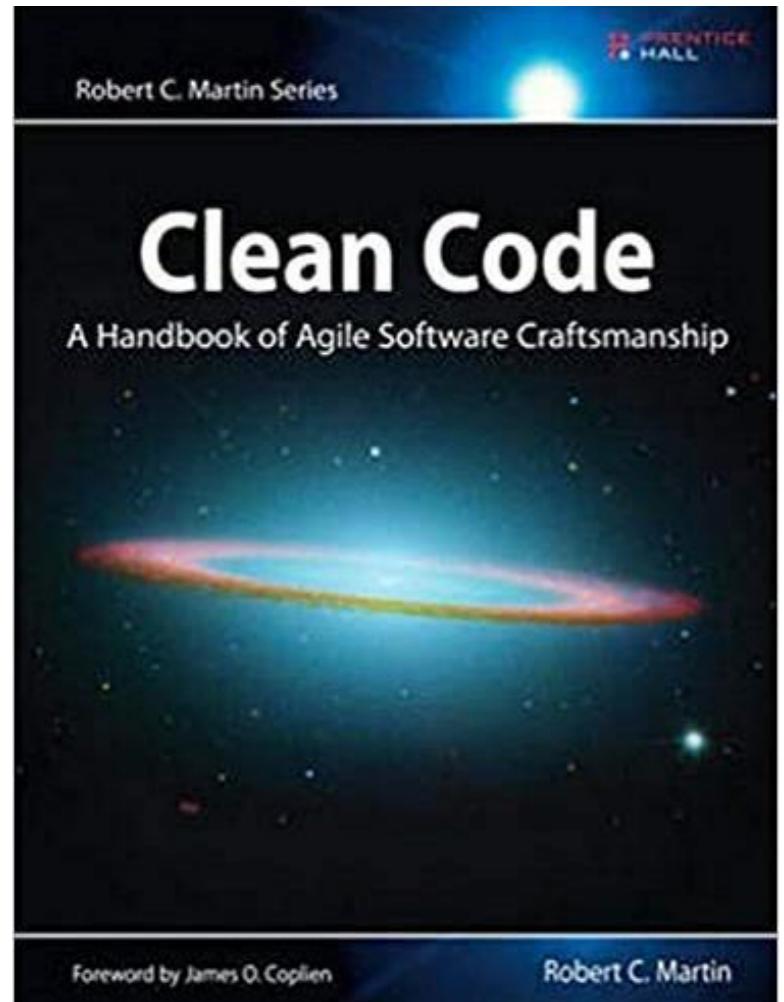
WANT TO DO IT



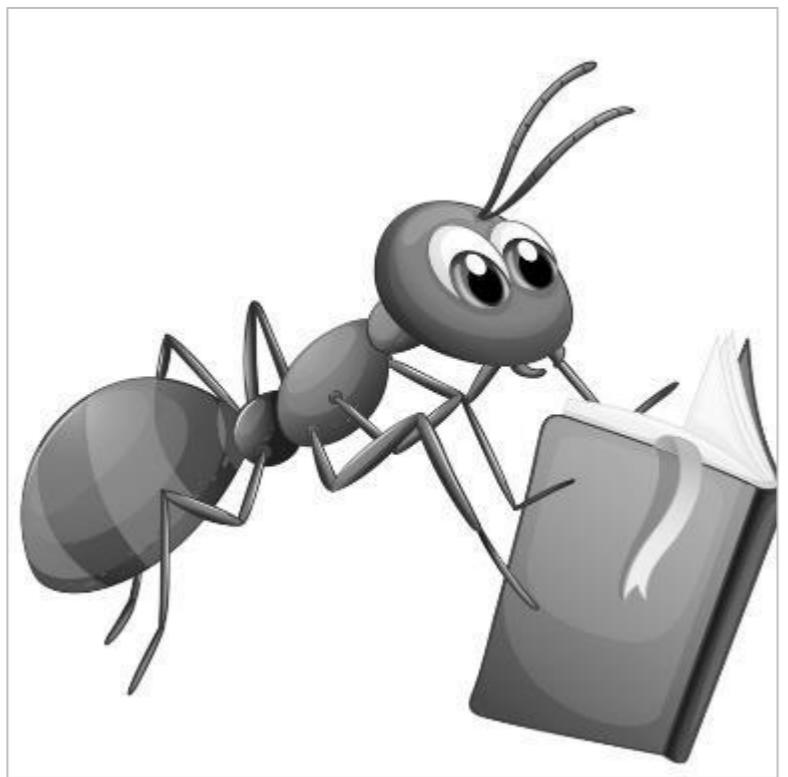
Q & A

ANNEXURE

GOOD READS



principles
patterns
practices
heuristics



+

Grind that
knowledge
into muscle
memory



=



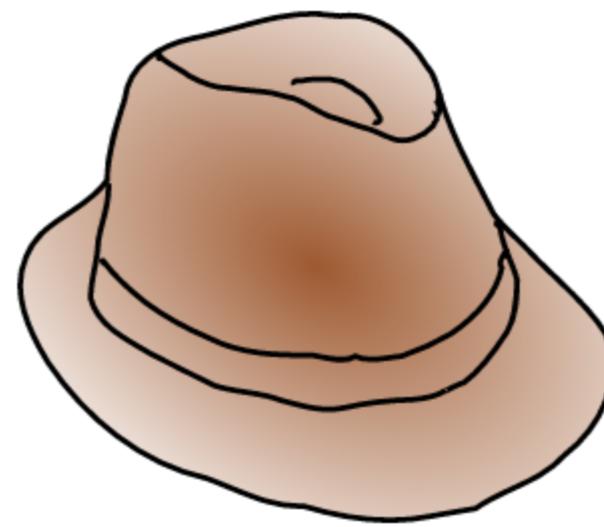
KNOWLEDGE

WORK

CRAFTSMANSHIP

“HONESTY IN SMALL THINGS
IS NOT A SMALL THING.”

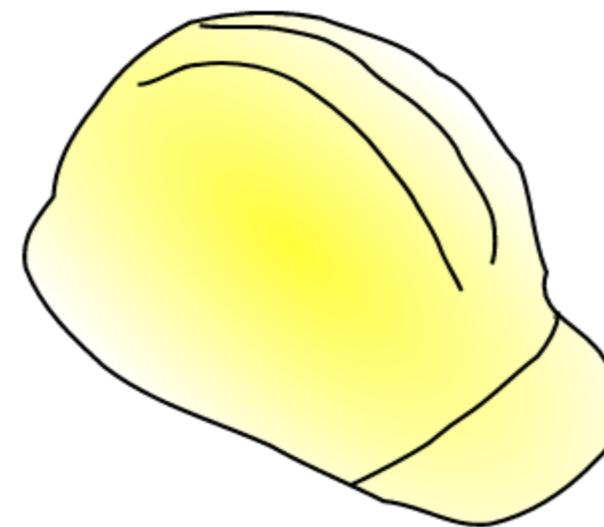
- Danish Saw



Refactoring

When refactoring every change you make is a small behavior-preserving change. You only refactor with green tests, and any test failing indicates a mistake. By stringing together a series of small changes like this you can move more quickly and with less risk because you shouldn't get trapped in debugging.

During programming you may swap frequently between hats, perhaps every couple of minutes. But...



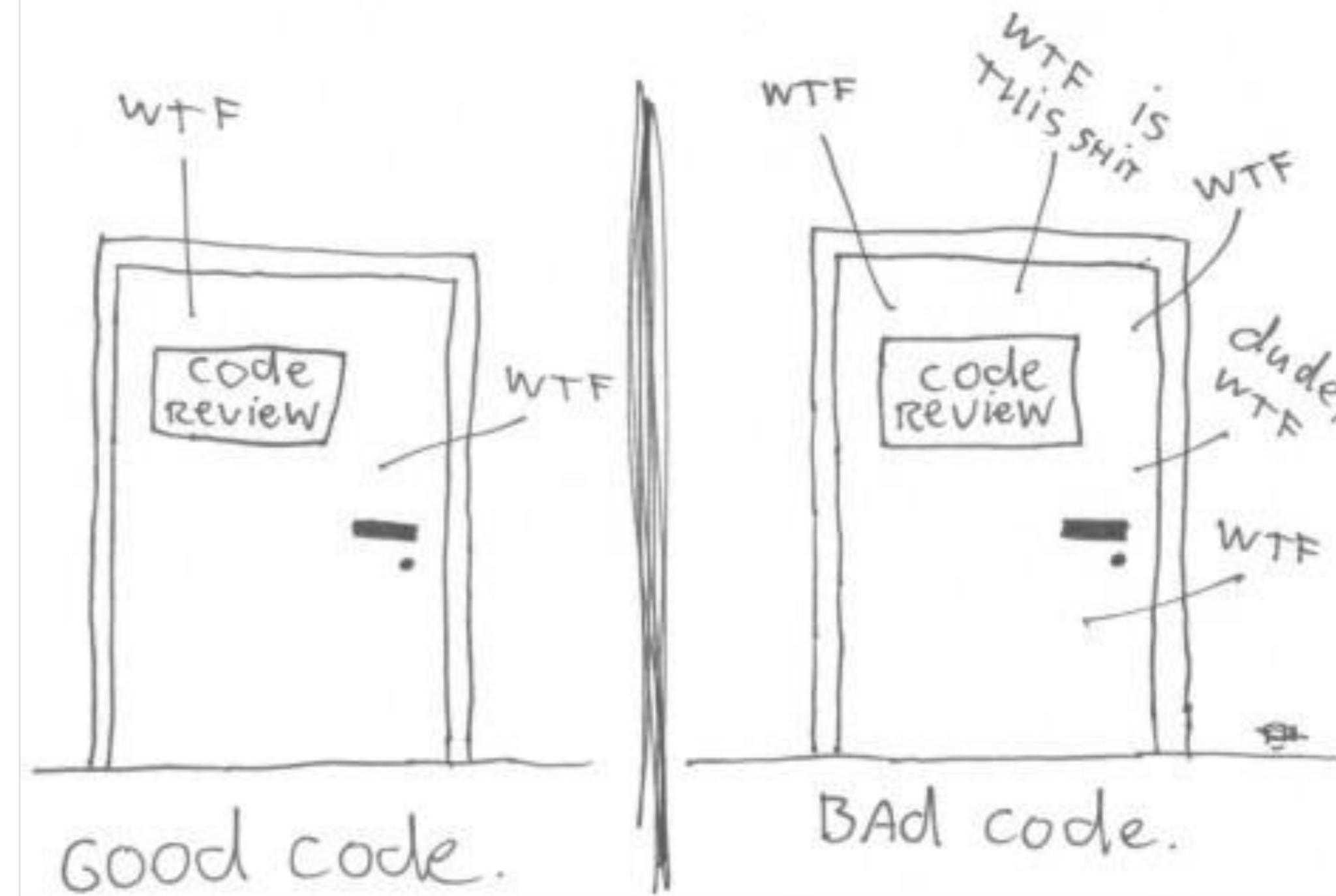
Adding Function

Any other change to the code is adding function. You will add new tests and break existing tests. You aren't confined to behavior-preserving changes (but it's wise to keep changes small and return to green tests swiftly).

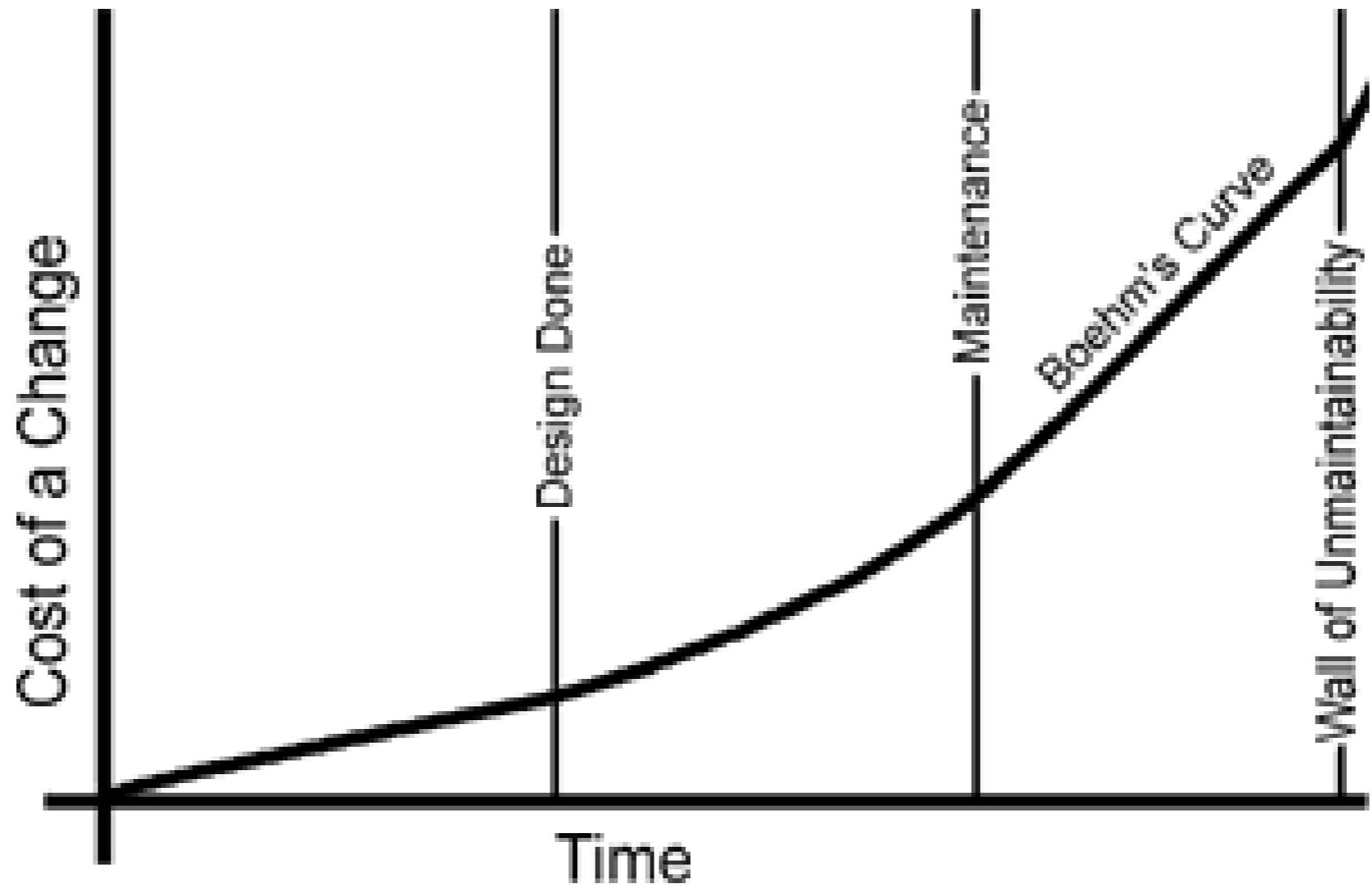
You can only wear one hat at a time

UNIT OF CODE QUALITY

The ONLY VALID MEASUREMENT
OF CODE QUALITY: WTFs/minute



(c) 2008 Focus Shift/OSNews/Thom Holwerda - <http://www.osnews.com/comics>

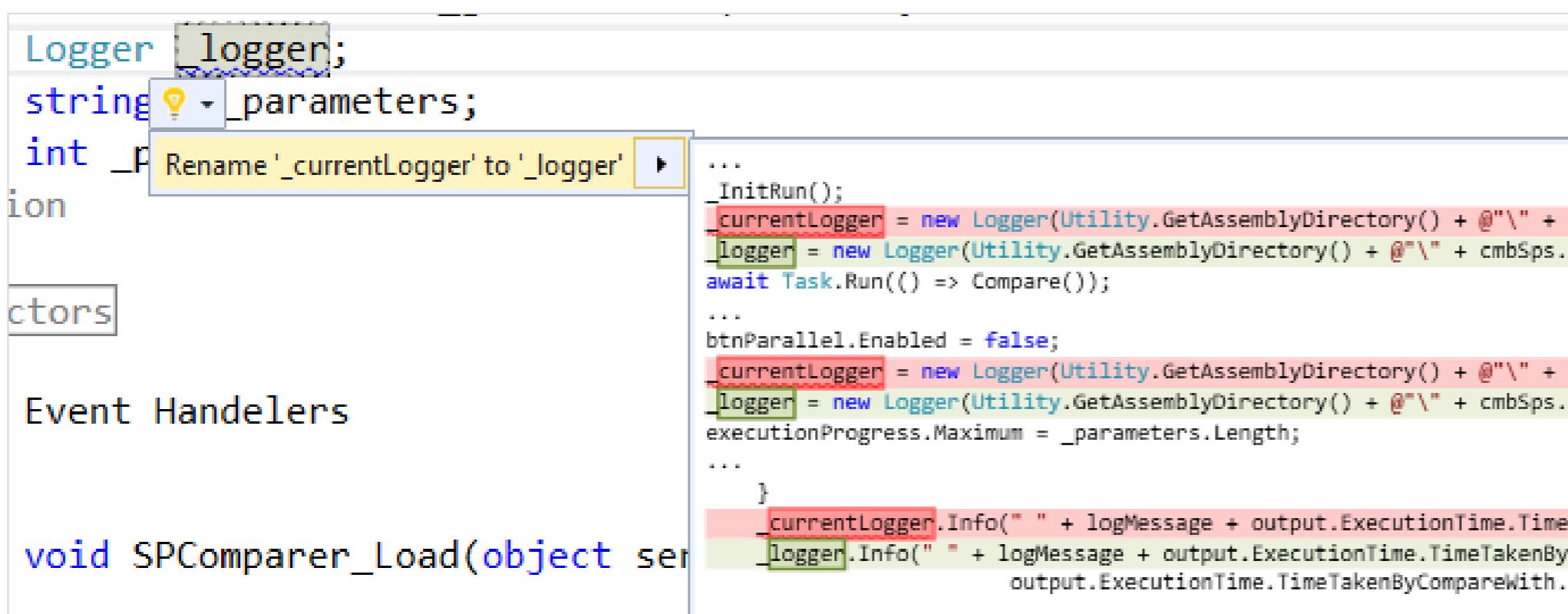


When we proclaim the design is done and accept no more changes

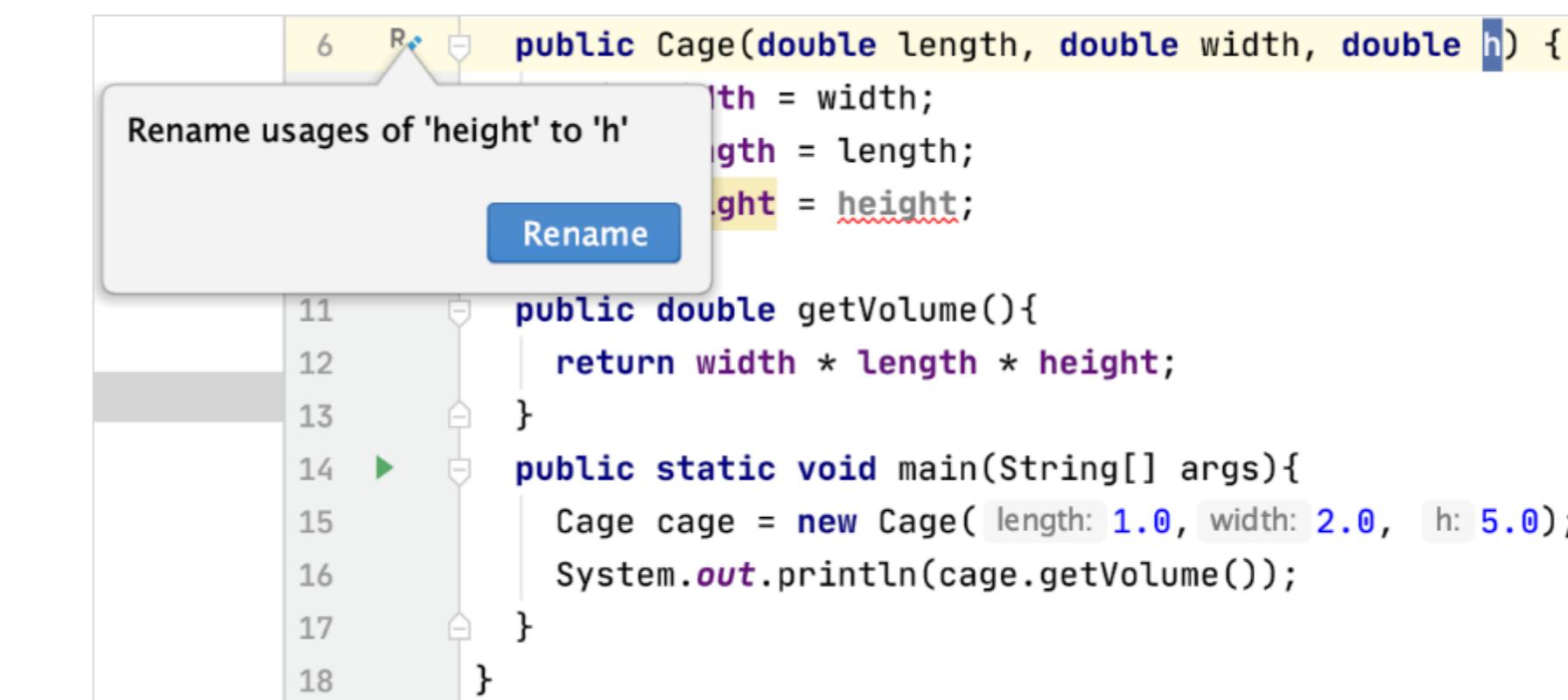
When we move the system into maintenance and change the team's process

THERE ARE ONLY TWO HARD THINGS IN
COMPUTER SCIENCE:
CACHE INVALIDATION AND NAMING THINGS

MOST OF THE IDES GIVES YOU VERY
SAFE WAY TO RENAME
VARIABLES, CLASSES, FUNCTIONS
ESPECIALLY PRIVATE STUFF IS
COMPLETELY SAGE



A screenshot of an IDE showing a code editor with C# syntax. A tooltip labeled 'Rename'_currentLogger to '_logger' is displayed over a line of code. The code snippet shows a logger being instantiated and assigned to a field named '_currentLogger'. The tooltip contains the new name '_logger'.



A screenshot of an IDE showing a code editor with Java syntax. A tooltip labeled 'Rename usages of 'height' to 'h'' is displayed over a line of code. The code snippet shows a class 'Cage' with a constructor taking 'length', 'width', and 'height' parameters. The tooltip contains the new name 'h'.

COMPREHENSION REFACTORING

Manifesto for Software Craftsmanship

Raising the bar.

Aspiring Software Craftsmen we are raising the bar of professional software development by practicing it and helping others learn the craft. Through this work we have come to value:

Not only working software,
but also **well-crafted software**

Not only responding to change,
but also **steadily adding value**

Not only individuals and interactions,
but also **a community of professionals**

Not only customer collaboration,
but also **productive partnerships**

That is, in pursuit of the items on the left we have found the items on the right to be indispensable.

© 2009, the undersigned.
this statement may be freely copied in any form,
but only in its entirety, without alteration.

MANIFESTO FOR SOFTWARE CRAFTSMANSHIP

EXTREME PROGRAMMING PRACTICES

Group	Practices
Feedback	<ul style="list-style-type: none">✓ Test-Driven Development✓ The Planning Game✓ On-site Customer✓ Pair Programming
Continual Process	<ul style="list-style-type: none">✓ Continuous Integration✓ Code Refactoring✓ Small Releases
Code understanding	<ul style="list-style-type: none">✓ Simple Design✓ Collective Code Ownership✓ System Metaphor✓ Coding Standards
Work conditions	<ul style="list-style-type: none">✓ 40-Hour Week

XP has simple rules that are based on **5 values**

COMMUNICATION

Everyone on a team works jointly at every stage of the project

SIMPLICITY

Developers strive to write simple code bringing more value to a product, as it saves time and efforts.

FEEDBACK

Team members deliver software frequently, get feedback about it, and improve a product according to the new requirements.

RESPECT

Every person assigned to a project contributes to a common goal.

COURAGE

Programmers objectively evaluate their own results without making excuses and are always ready to respond to changes.