Clean Code 2: More Tips for Writing Clear and Concise Code

Ben Rand

Director of IT, Senior Developer

Job Industrial Services, Inc.

AUTODESK. UNIVERSITY

About the speaker

Ben Rand

Ben has been using AutoCAD since Release 12. He learned to program using LISP in AutoCAD, worked his way up through VBA, VB6 and VB.NET, and now spends most of his days programming in C# (occasionally still in AutoCAD!), and dabbling in other languages such as Java, Ruby and Python. He has worked in the Industrial Engineering field for more than 18 years as a CAD Manager, developer and IT Director. In 2013, he was the 2013 Top DAUG overall winner at AU, and he served a mentor for the AutoCAD Mentor All-Star team. He has been presenting at Autodesk University since 2015, and made the Top Speaker list at AU 2017 and 2018.

Ben is the proud father of four children and enjoys reading and playing a variety of sports including pickleball, volleyball, and tennis. In 2018, Ben’s USTA (United States Tennis Association) men's league teams won the National Championship in 18 and over, and placed 2nd in 40 and over.

Course Objectives

NM

NAME MEANINGFULLY

Name classes, methods and variables meaningfully

CF

COHESIVE FOCUSED

Write more cohesive, focused methods and classes

UT

UNIT TESTS

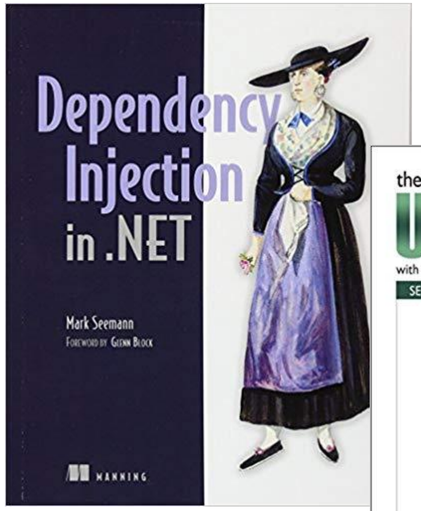
Utilize unit tests to support refactoring towards clean code

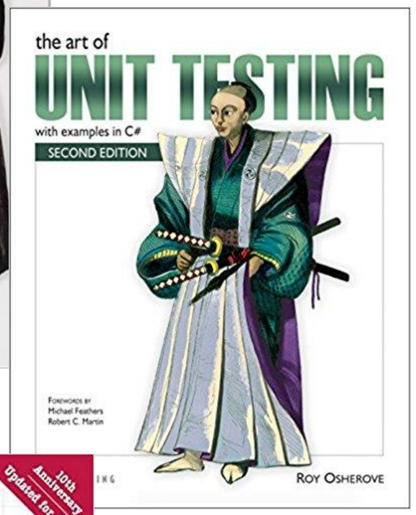
DI

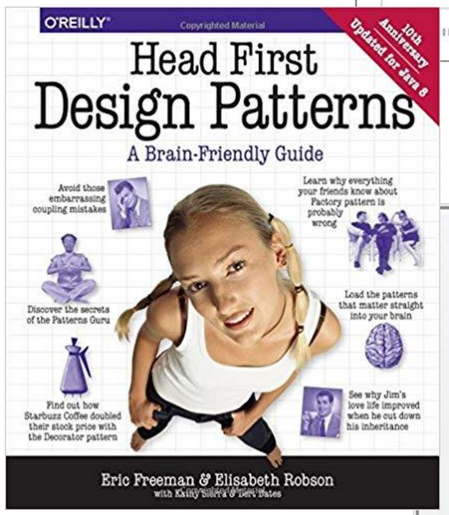
DEPENDENCY INJECTION

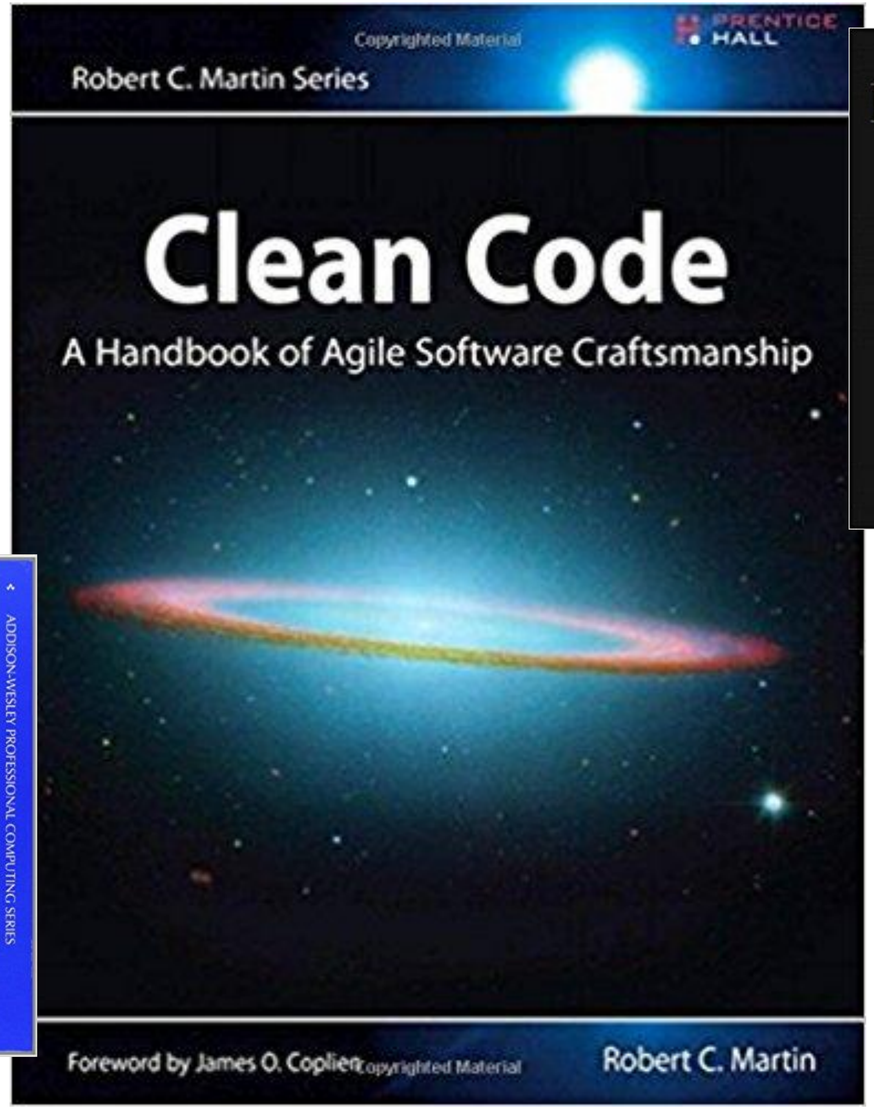
Learn how to use dependency injection to create more flexible apps

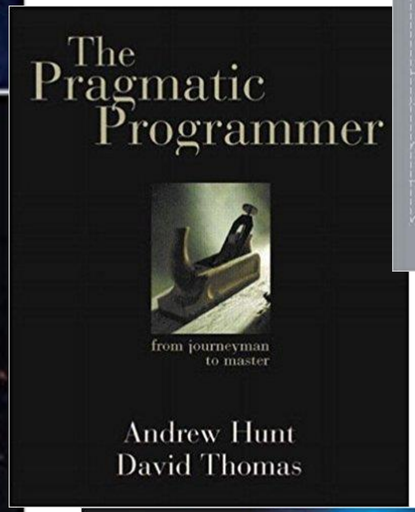
This class inspired by:

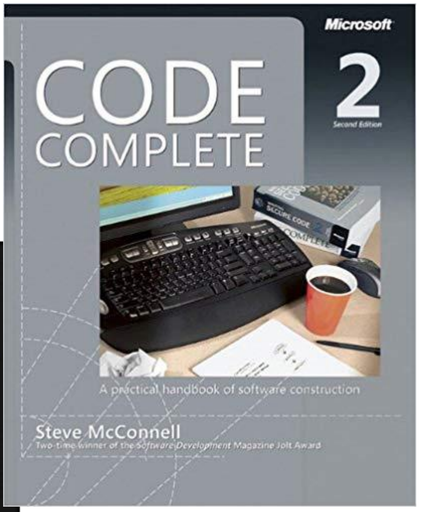




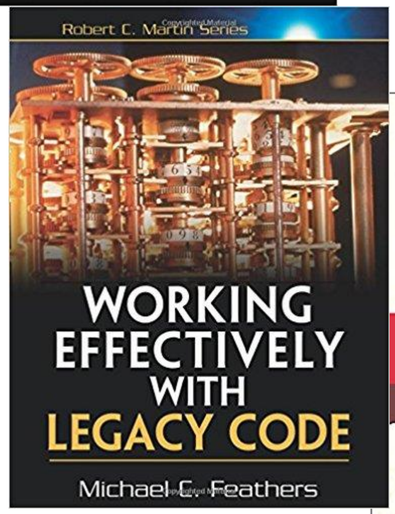


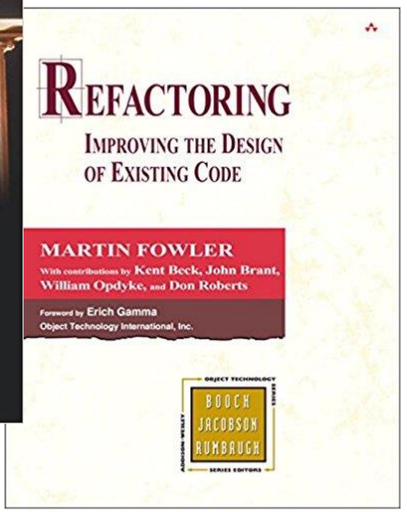




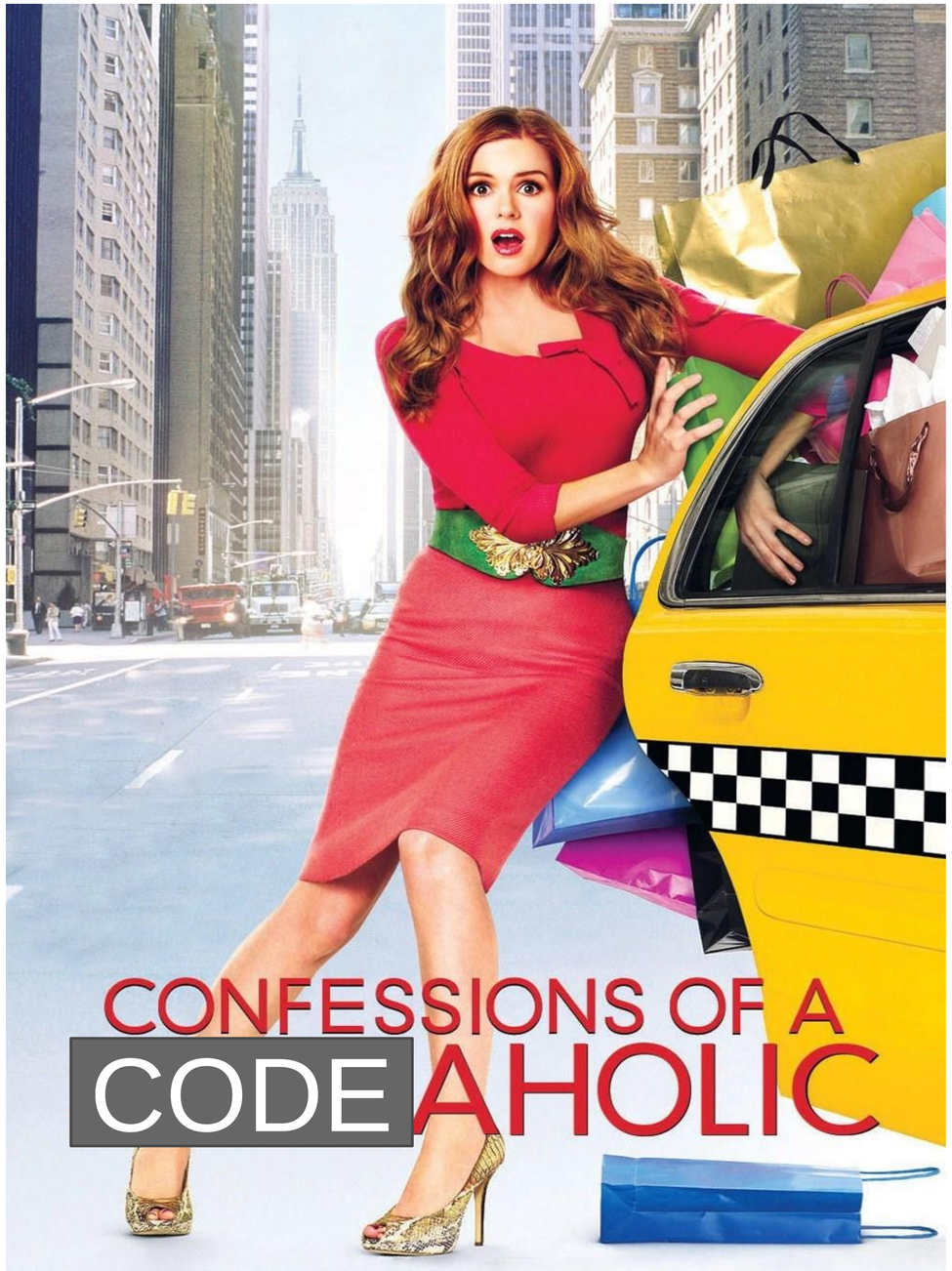


Mentoring others





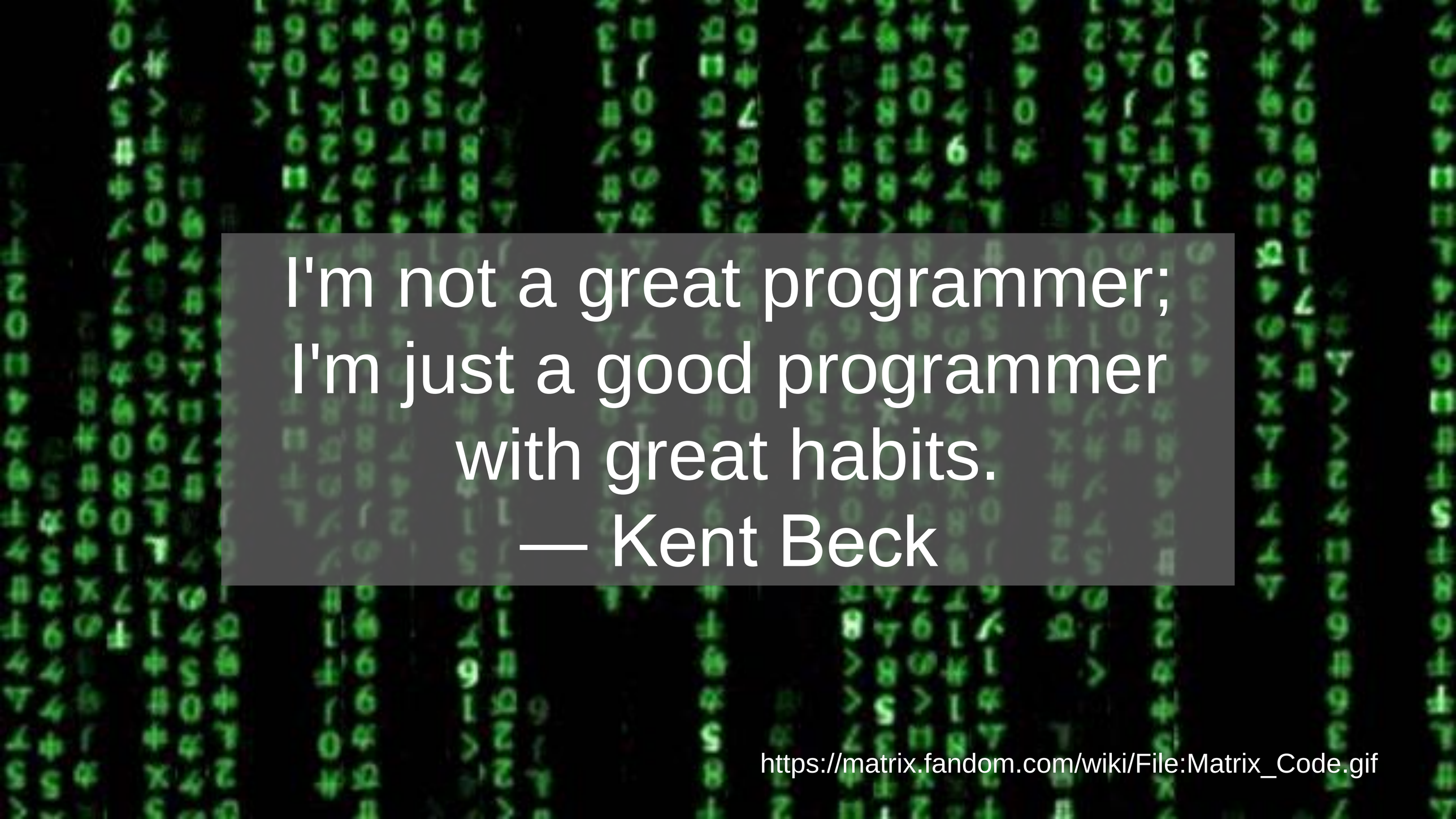
My own bad code



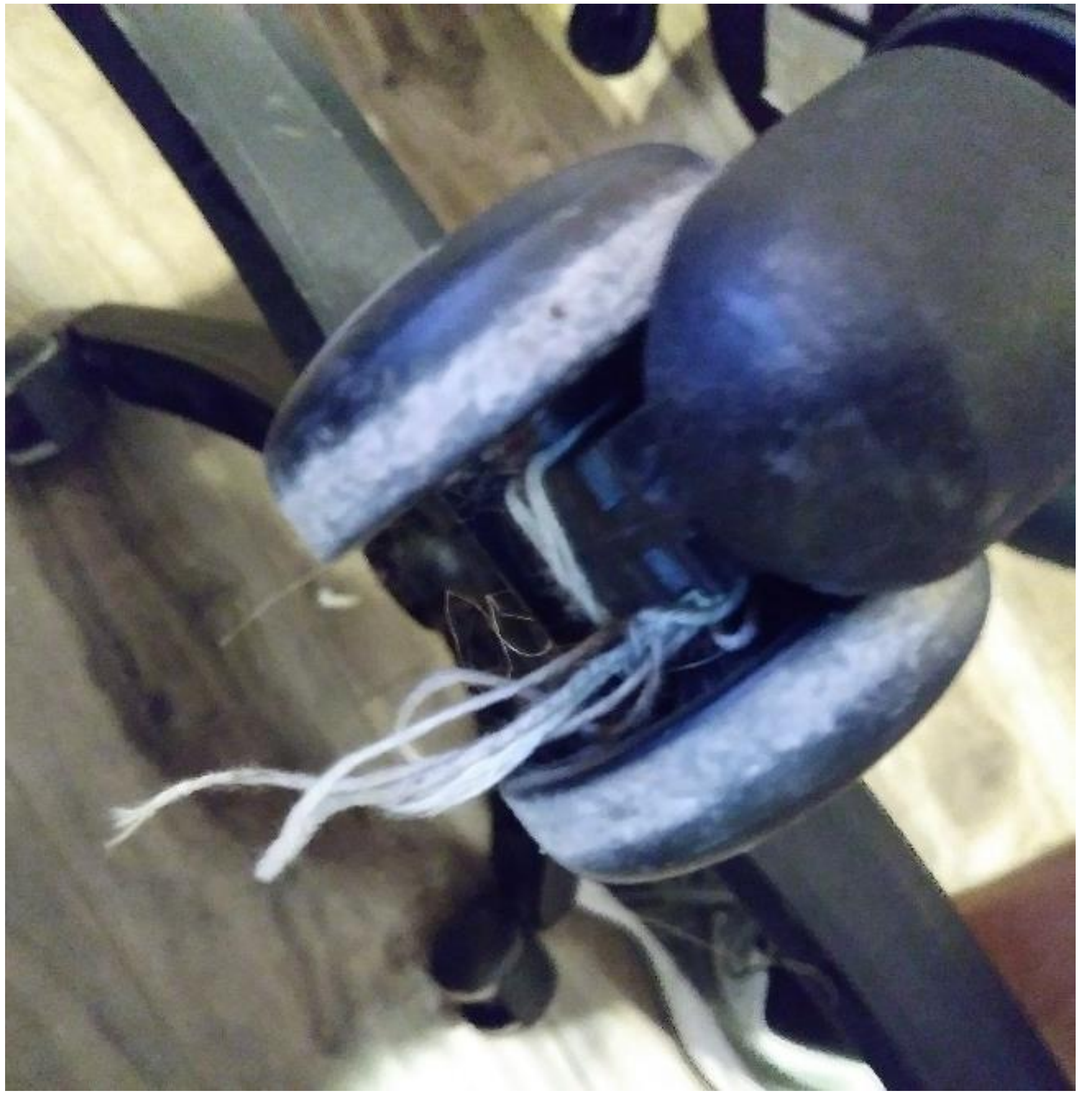
http://beamiller.wikia.com/wiki/Confessions_of_a_Shopaholic

Confession: I have written bad code

- Professionally
- In many languages:
  - LISP
  - VBA/VB6
  - VB.NET
  - C#
  - Java
  - Ruby
  - Javascript
- I still write bad code, but I’m getting better at:
  - Recognizing it when I do
  - Doing something about it



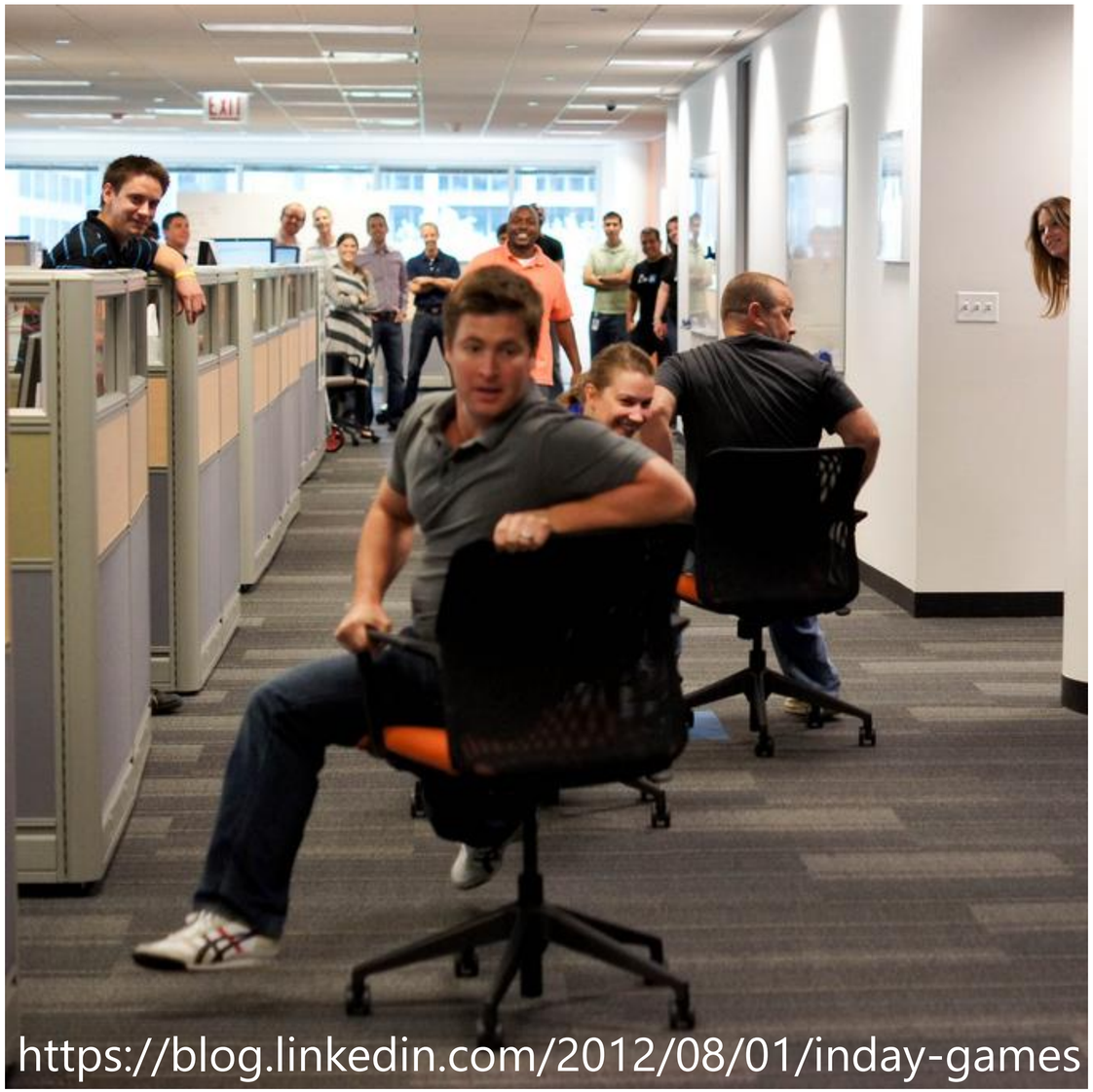
https://matrix.fandom.com/wiki/File:Matrix_Code.gif



Why is Bad Code…bad?

- Bad code **works** BUT
- *Lacks unit tests*
- Is hard to read
- Is hard to reason about
- Is hard to maintain
- Tries to do too much
- Uses confusing names
- Relies on magic numbers/strings
- Is poorly formatted

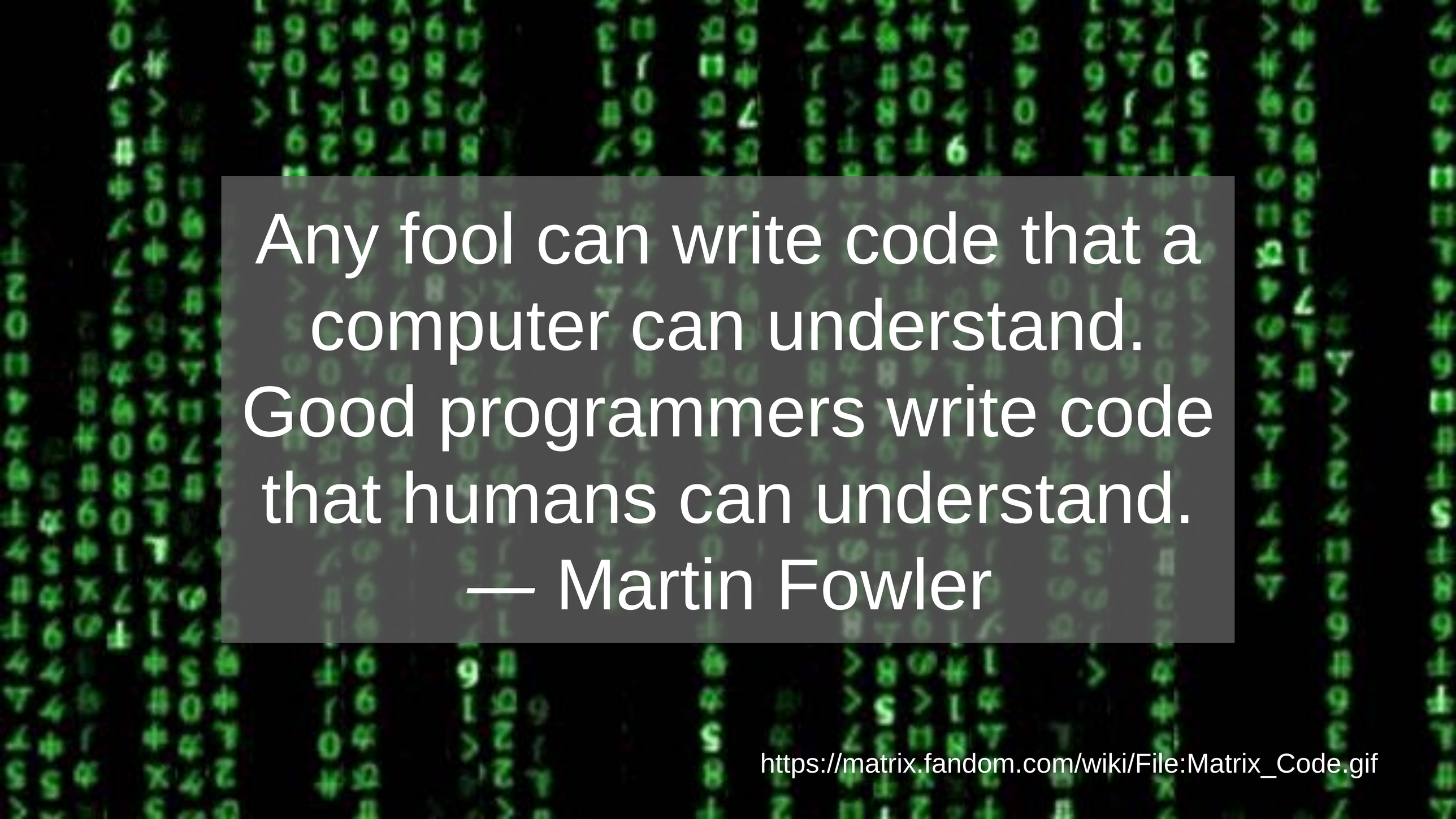
You have to work extra hard to work with bad code—over and over again.



https://blog.linkedin.com/2012/08/01/inday-games

Why is Clean Code better?

- Clean code **works** AND
- *Has unit tests*
- Is easy to read
- Is easy to reason about
- Is easy to maintain
- Is easy to extend
- Is easy to understand several weeks, months or years(!) from now


Any fool can write code that a computer can understand. Good programmers write code that humans can understand.
— Martin Fowler
https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Naming – Classes, Methods, Variables

- ***Reveal intent***
- ***Avoid disinformation***
  - Avoid “magic” numbers and strings
- ***Use longer, descriptive names vs short, cryptic names***
  - Avoid single letter names
  - Pronounceable
  - Avoid “Hungarian notation” prefixes
- **Nouns for classes**
- **Verb/verb phrases for methods**

Naming – Reveal Intent/Avoid Disinformation

```
/**
 * Checks to see if a credit card number is valid.
 * <p>Rule 1: Double every 2nd digit from right to left, starting with second digit from right
 *            If doubling the digit results in two-digit number, add the two digits to get a
 *            single digit number.</p>
 * <p>Rule 2: Add together all the resulting single digit numbers from Rule 1.</p>
 * <p>Rule 3: Add every other digit from right to left starting from the right-most digit.</p>
 * <p>Rule 4: Sum the result from step 2 and step 3.
 *            If the result is divisible by 10, card number is valid, otherwise it is invalid.</p>
 **/

if(valid(numbersDoubledSum(doubler(numSeperator(userInput))),
    numberSum(numSeperator(userInput))) != false)
    {
        //do something
    }
```

Naming – Reveal Intent

```
if(valid(numbersDoubledSum(doubler(numSeperator(userInput))),
    numberSum(numSeperator(userInput))) != false)
    { //do something }
```

```
public class CreditCardValidator {
    public CreditCardValidator(long ccNumber)
    {
        _ccNumber = ccNumber;
    }
    public ArrayList<Integer> getDigits() {...}
    public int getOddDigitsCheckSum(ArrayList<Integer> digits) { ... }

    public int getEvenDigitsCheckSum(ArrayList<Integer> digits) { ... }

    public boolean isValid()
    {
        ArrayList<Integer> digits = getDigits();
        int oddCheckSum = getOddDigitsCheckSum(digits);
        int evenCheckSum = getEvenDigitsCheckSum(digits);

        int checkSum = (oddCheckSum + evenCheckSum);
        int remainder = checkSum % 10;
        return remainder == 0;
    }
}
```

Naming – Pronounceable and Descriptive

```
Using tr As Transaction = db.TransactionManager.StartTransaction
    'Do stuff
    tr.Commit()
End Using

Using tran As Transaction = db.TransactionManager.StartTransaction
    'Do stuff
    tran.Commit()
End Using

Using transaction As Transaction = db.TransactionManager.StartTransaction
    'Do stuff
    transaction.Commit()
End Using
```

Naming – Avoid Identifying Object Type in Name

```
TextBox txtFirstName;
TextBox txtLastName;
ComboBox cboDepartments;
```

```
TextBox firstName;
TextBox lastName;
ComboBox departments;
```

```
IList<Employee> employeeList;
```

```
IList<Employee> employees;
```

Naming – “Magic” Numbers and Strings

```
(setvar "osmode" 512) ;set near on
(setq "osmode" 29) ;set end+center+node+quadrant on


(setq END_SNAP 1
      MID_SNAP 2
      CENTER_SNAP 4
      NODE_SNAP 8
      QUADRANT_SNAP 16)

(setq "osmode" (+ END_SNAP                NODE_SNAP QUADRANT_SNAP))
```

Naming – “Magic” Numbers and Strings

```
(cond ((and (= block "CS-2")(< 8.1 (distof cadworxalp
  (alert "Not suggested for this size pipe!")))
```

Appeared 20+ times!

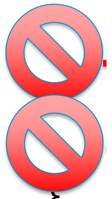
```
(setq NOT_SUGGESTED_MSG "Not suggested for this size pipe!")
(cond ((and (= block "CS-2")(< 8.1 (distof cadworxalpha 4)))
  (alert NOT_SUGGESTED_MSG)))
```

Classes

- **Can I describe its purpose in 25 words or less, without using: if, and, or, but?**
- **Should be short**
- **Use nouns for names**
- **SOLID:** https://en.wikipedia.org/wiki/SOLID
  - Single responsibility principle (SRP)—single reason to change
  - Open/closed principle—open for extension, closed for modification
  - Liskov substitution principle—if S is subtype of T, T and S should be interchangeable
  - Interface segregation principle—smaller interfaces are better
  - Dependency inversion principle—code to abstractions

Classes – Short, Single Responsibility

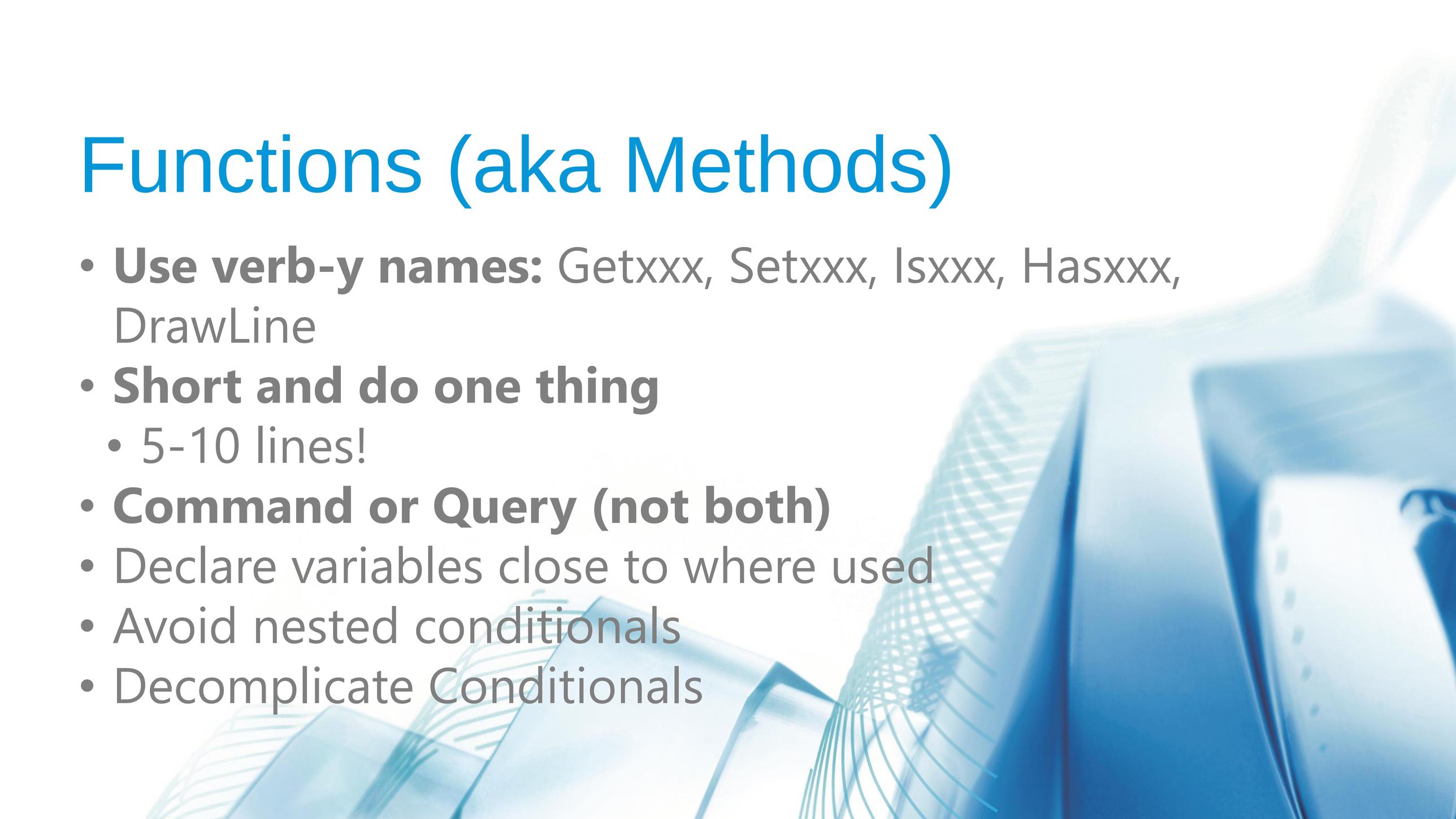
```
public class CreditCardValidator {
    public CreditCardValidator(long ccNumber)
    {
        _ccNumber = ccNumber;
    }
    public ArrayList<Integer> getDigits() {...}

    public int getOddDigitsCheckSum(ArrayList<Integer> digits) { ... }

    public int getEvenDigitsCheckSum(ArrayList<Integer> digits) { ... }

    public boolean isValid() { ... }
```



```
}
```

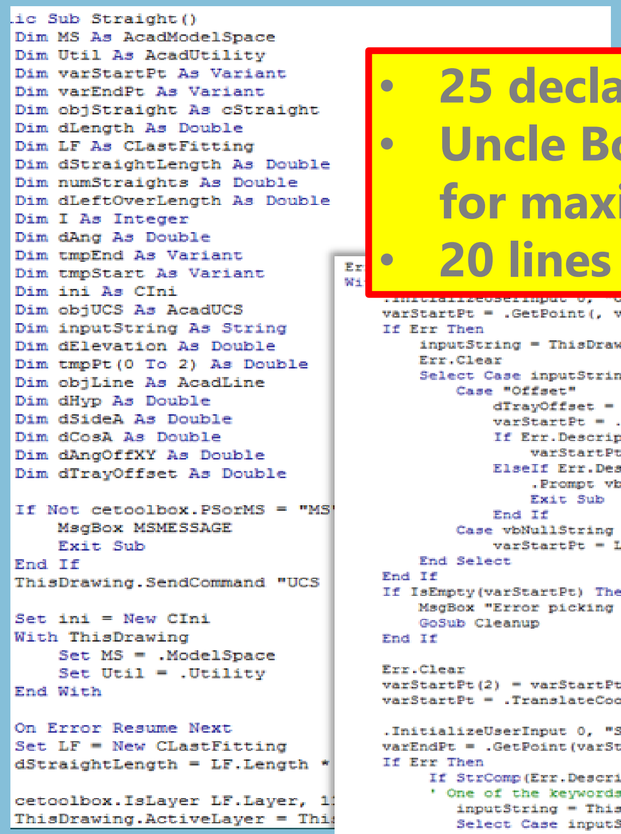
Functions (aka Methods)

- **Use verb-y names:** Getxxx, Setxxx, Isxxx, Hasxxx, DrawLine
- **Short and do one thing**
  - 5-10 lines!
- **Command or Query (not both)**
- Declare variables close to where used
- Avoid nested conditionals
- Decomplicate Conditionals

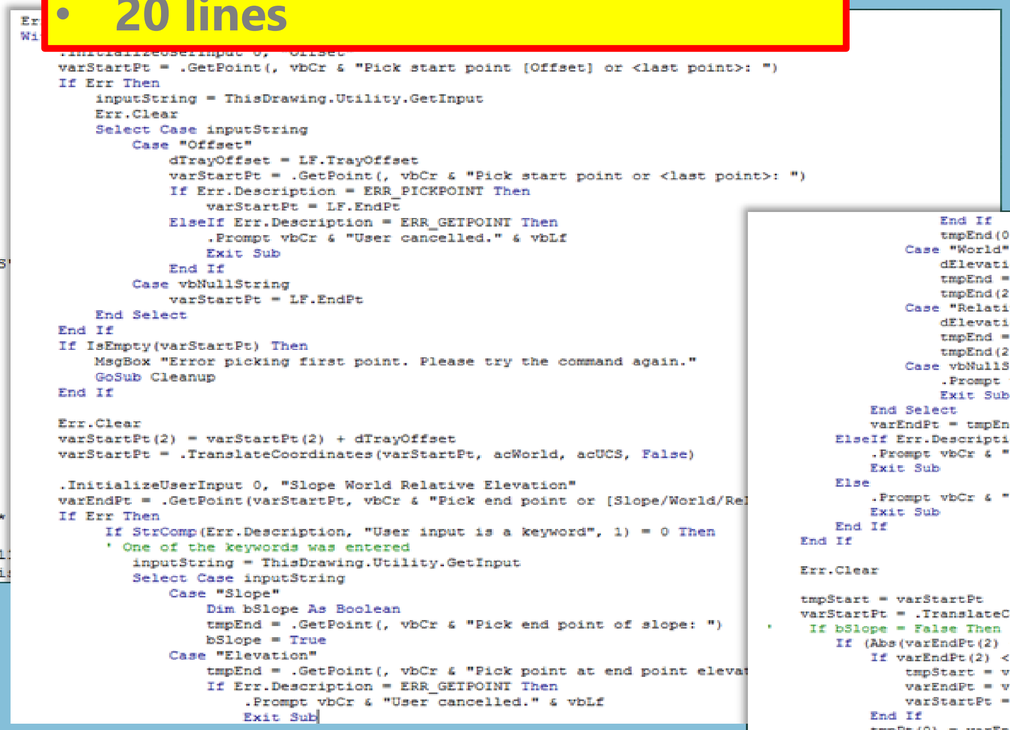


Functions – Short and Do One Thing

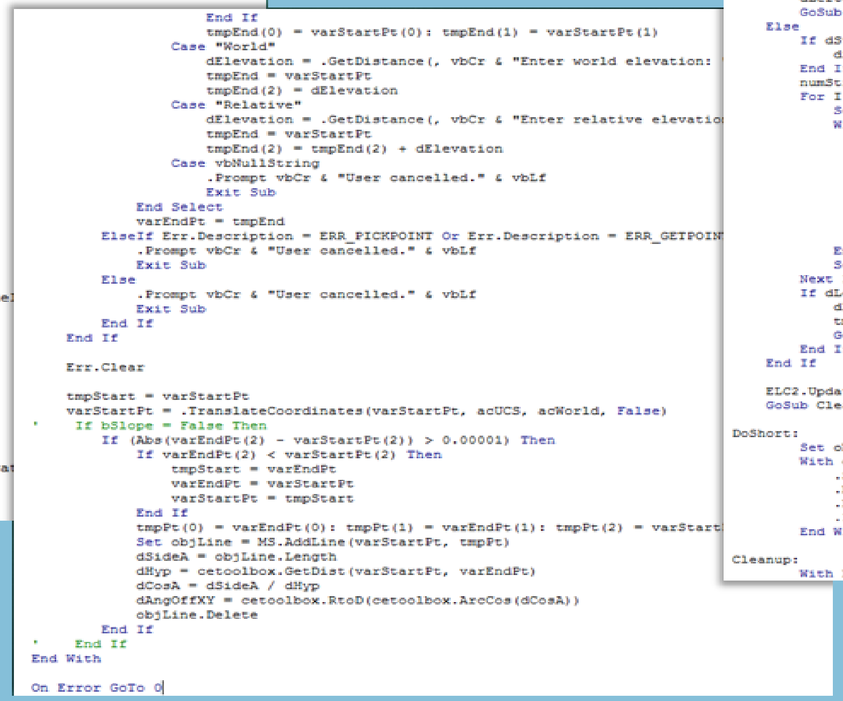
- **25 declared variables**
- **Uncle Bob's recommendation for maximum method length?**
- **20 lines**



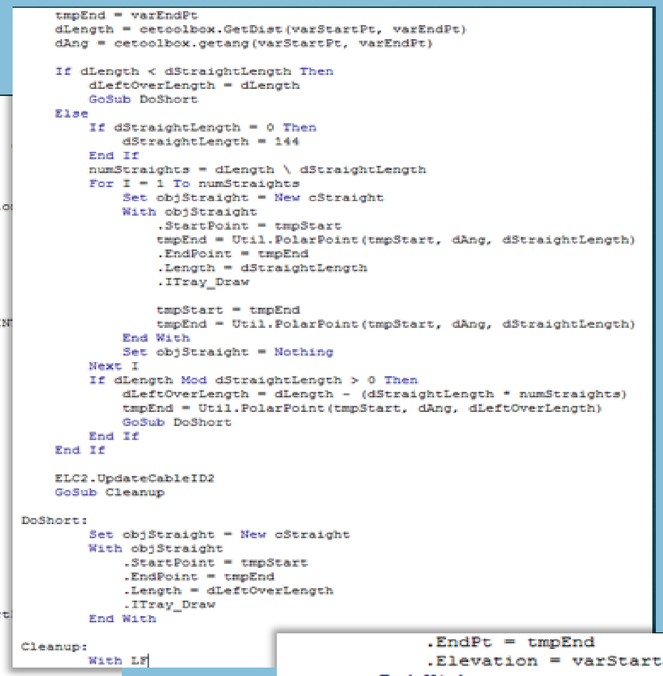
```
ic Sub Straight()
Dim MS As AcadModelSpace
Dim Util As AcadUtility
Dim varStartPt As Variant
Dim varEndPt As Variant
Dim objStraight As cStraight
Dim dLength As Double
Dim LF As CLastFitting
Dim dStraightLength As Double
Dim numStraights As Double
Dim dLeftOverLength As Double
Dim I As Integer
Dim dAng As Double
Dim tmpEnd As Variant
Dim tmpStart As Variant
Dim ini As CIni
Dim objUCS As AcadUCS
Dim inputString As String
Dim dElevation As Double
Dim tmpPt(0 To 2) As Double
Dim objLine As AcadLine
Dim dHyp As Double
Dim dSideA As Double
Dim dCosA As Double
Dim dAngOffXY As Double
Dim dTrayOffset As Double

If Not cetoolbox.PSorMS = "MS"
    MsgBox MSMESSAGE
    Exit Sub
End If
ThisDrawing.SendCommand "UCS

Set ini = New CIni
With ThisDrawing
    Set MS = .ModelSpace
    Set Util = .Utility
End With

On Error Resume Next
Set LF = New CLastFitting
dStraightLength = LF.Length *

cetoolbox.IsLayer LF.Layer, 1
ThisDrawing.ActiveLayer = This
```



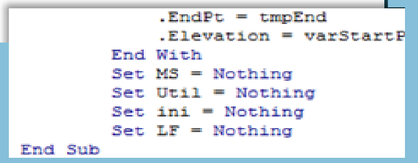
```
varStartPt = .GetPoint(, vbCr & "Pick start point [Offset] or <last point>: ")
If Err Then
    inputString = ThisDrawing.Utility.GetInput
    Err.Clear
    Select Case inputString
        Case "Offset"
            dTrayOffset = LF.TrayOffset
            varStartPt = .GetPoint(, vbCr & "Pick start point or <last point>: ")
            If Err.Description = ERR_PICKPOINT Then
                varStartPt = LF.EndPt
            ElseIf Err.Description = ERR_GETPOINT Then
                .Prompt vbCr & "User cancelled." & vbLf
                Exit Sub
            End If
        Case vbNullString
            varStartPt = LF.EndPt
    End Select
End If
If IsEmpty(varStartPt) Then
    MsgBox "Error picking first point. Please try the command again."
    GoSub Cleanup
End If

Err.Clear
varStartPt(2) = varStartPt(2) + dTrayOffset
varStartPt = .TranslateCoordinates(varStartPt, acWorld, acUCS, False)

.InitializeUserInput 0, "Slope World Relative Elevation"
varEndPt = .GetPoint(varStartPt, vbCr & "Pick end point or [Slope/World/Rel
If Err Then
    If StrComp(Err.Description, "User input is a keyword", 1) = 0 Then
    ' One of the keywords was entered
        inputString = ThisDrawing.Utility.GetInput
        Select Case inputString
            Case "Slope"
                Dim bSlope As Boolean
                tmpEnd = .GetPoint(, vbCr & "Pick end point of slope: ")
                bSlope = True
            Case "Elevation"
                tmpEnd = .GetPoint(, vbCr & "Pick point at end point elevat
                If Err.Description = ERR_GETPOINT Then
                    .Prompt vbCr & "User cancelled." & vbLf
                    Exit Sub
```



```
                End If
                tmpEnd(0) = varStartPt(0): tmpEnd(1) = varStartPt(1)
            Case "World"
                dElevation = .GetDistance(, vbCr & "Enter world elevation:
                tmpEnd = varStartPt
                tmpEnd(2) = dElevation
            Case "Relative"
                dElevation = .GetDistance(, vbCr & "Enter relative elevatio
                tmpEnd = varStartPt
                tmpEnd(2) = tmpEnd(2) + dElevation
            Case vbNullString
                .Prompt vbCr & "User cancelled." & vbLf
                Exit Sub
        End Select
        varEndPt = tmpEnd
    ElseIf Err.Description = ERR_PICKPOINT Or Err.Description = ERR_GETPOINT
        .Prompt vbCr & "User cancelled." & vbLf
        Exit Sub
    Else
        .Prompt vbCr & "User cancelled." & vbLf
        Exit Sub
    End If
End If

Err.Clear

tmpStart = varStartPt
varStartPt = .TranslateCoordinates(varStartPt, acUCS, acWorld, False)
'   If bSlope = False Then
    If (Abs(varEndPt(2) - varStartPt(2)) > 0.00001) Then
        If varEndPt(2) < varStartPt(2) Then
            tmpStart = varEndPt
            varEndPt = varStartPt
            varStartPt = tmpStart
        End If
        tmpPt(0) = varEndPt(0): tmpPt(1) = varEndPt(1): tmpPt(2) = varStart
        Set objLine = MS.AddLine(varStartPt, tmpPt)
        dSideA = objLine.Length
        dHyp = cetoolbox.GetDist(varStartPt, varEndPt)
        dCosA = dSideA / dHyp
        dAngOffXY = cetoolbox.RtoD(cetoolbox.ArcCos(dCosA))
        objLine.Delete
    End If
'   End If
End With

On Error GoTo 0
```



```
    tmpEnd = varEndPt
    dLength = cetoolbox.GetDist(varStartPt, varEndPt)
    dAng = cetoolbox.getang(varStartPt, varEndPt)

    If dLength < dStraightLength Then
        dLeftOverLength = dLength
        GoSub DoShort
    Else
        If dStraightLength = 0 Then
            dStraightLength = 144
        End If
        numStraights = dLength \ dStraightLength
        For I = 1 To numStraights
            Set objStraight = New cStraight
            With objStraight
                .StartPoint = tmpStart
                tmpEnd = Util.PolarPoint(tmpStart, dAng, dStraightLength)
                .EndPoint = tmpEnd
                .Length = dStraightLength
                .ITray_Draw

                tmpStart = tmpEnd
                tmpEnd = Util.PolarPoint(tmpStart, dAng, dStraightLength)
            End With
            Set objStraight = Nothing
        Next I
        If dLength Mod dStraightLength > 0 Then
            dLeftOverLength = dLength - (dStraightLength * numStraights)
            tmpEnd = Util.PolarPoint(tmpStart, dAng, dLeftOverLength)
            GoSub DoShort
        End If
    End If

    ELC2.UpdateCableID2
    GoSub Cleanup

DoShort:
        Set objStraight = New cStraight
        With objStraight
            .StartPoint = tmpStart
            .EndPoint = tmpEnd
            .Length = dLeftOverLength
            .ITray_Draw
        End With

Cleanup:
        With LF
```



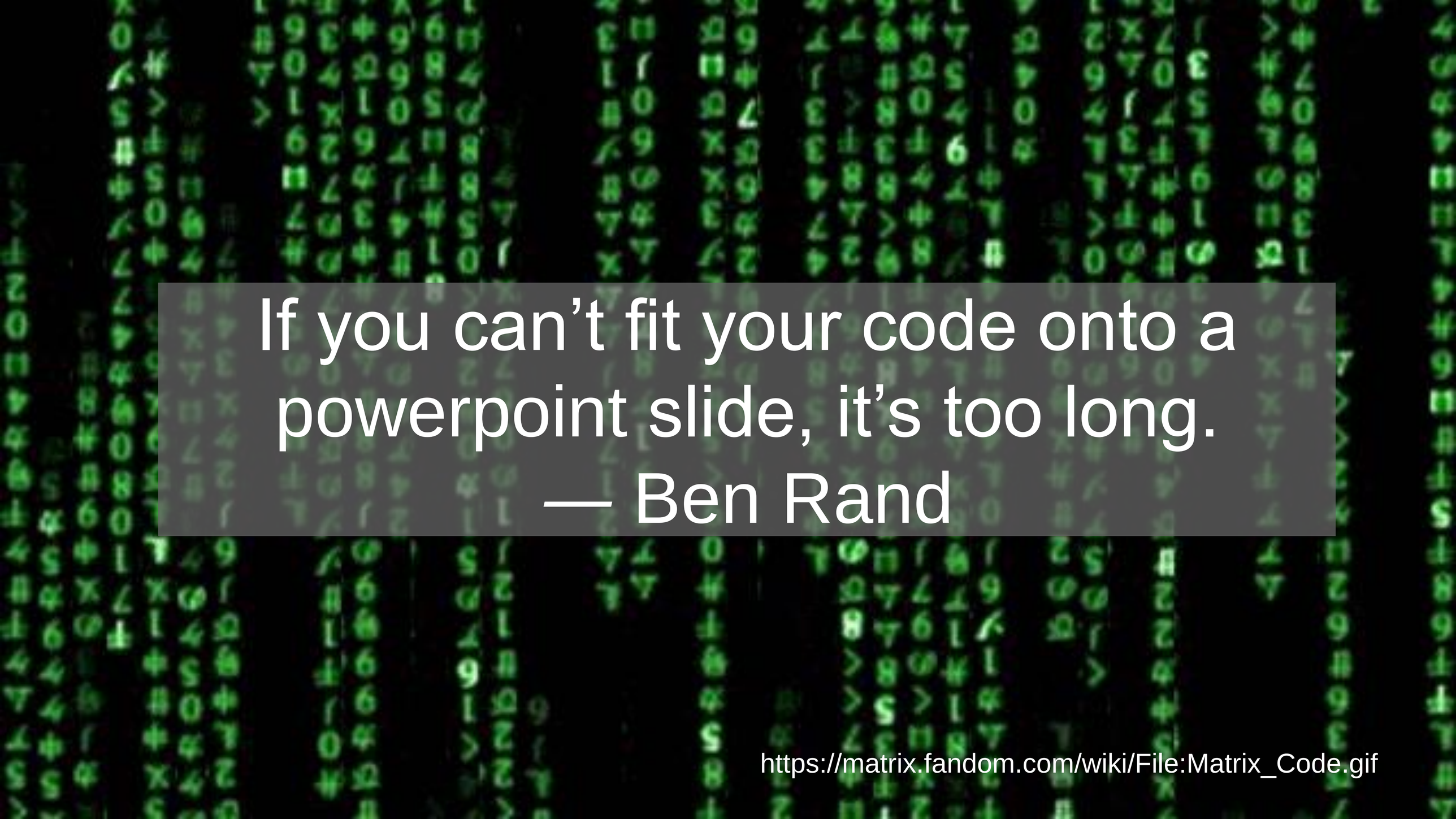
```
                .EndPt = tmpEnd
                .Elevation = varStartP
            End With
            Set MS = Nothing
            Set Util = Nothing
            Set ini = Nothing
            Set LF = Nothing
End Sub
```

Functions – Declare Variables Close to Usage

```
Dim dSideA As Double

Dim dAngOffXY As Double

If Not cetoolbox.PSorMS =
    MsgBox MSMESSAGE
    Exit Sub
End If
ThisDrawing.SendCommand "UCS

Set ini = New CIni
With ThisDrawing
    Set MS = .ModelSpace
    Set Util = .Utility
End With

On Error Resume Next
Set LF = New CLastFitting
dStraightLength = LF.Length *

cetoolbox.IsLayer LF.Layer, 1
ThisDrawing.ActiveLayer = This
```

```
            dTrayOffset = LF.TrayOffset
            varStartPt = .GetPoint(, vbCr & "Pick start point or <last point>: ")
            If Err.Description = ERR_PICKPOINT Then
                varStartPt = LF.EndPt
            ElseIf Err.Description = ERR_GETPOINT Then
                .Prompt vbCr & "User cancelled." & vbLf
                Exit Sub
            End If
        Case vbNullString
            varStartPt = LF.EndPt
    End Select
End If
If IsEmpty(varStartPt) Then
    MsgBox "Error picking first point. Please try the command again."
    GoSub Cleanup
End If

Err.Clear
varStartPt(2) = varStartPt(2) + dTrayOffset
varStartPt = .TranslateCoordinates(varStartPt, acWorld, acUCS, False)

.InitializeUserInput 0, "Slope World Relative Elevation"
varEndPt = .GetPoint(varStartPt, vbCr & "Pick end point or [Slope/World/Rel
If Err Then
    If StrComp(Err.Description, "User input is a keyword", 1) = 0 Then
    ' One of the keywords was entered
        inputString = ThisDrawing.Utility.GetInput
        Select Case inputString
            Case "Slope"
                Dim bSlope As Boolean
                tmpEnd = .GetPoint(, vbCr & "Pick end point of slope: ")
                bSlope = True
            Case "Elevation"
                tmpEnd = .GetPoint(, vbCr & "Pick point at end point eleva
                If Err.Description = ERR_GETPOINT Then
                    .Prompt vbCr & "User cancelled." & vbLf
                    Exit Sub
```

```
                End If
                tmpEnd(0) = varStartPt(0): tmpEnd(1) = varStar
            Case "World"
                dElevation = .GetDistance(, vbCr & "Enter worl
                tmpEnd = varStartPt
                tmpEnd(2) = dElevation
            Case "Relative"
                dElevation = .GetDistance(, vbCr & "Enter rela
                tmpEnd = varStartPt
                tmpEnd(2) = tmpEnd(2) + dElevation
            Case vbNullString
                .Prompt vbCr & "User cancelled." & vbLf
                Exit Sub
        End Select
        varEndPt = tmpEnd
    ElseIf Err.Description = ERR_PICKPOINT Or Err.Description
        .Prompt vbCr & "User cancelled." & vbLf
        Exit Sub
    Else
        .Prompt vbCr & "User cancelled." & vbLf
        Exit Sub
    End If
End If

Err.Clear

tmpStart = varStartPt
varStartPt = .TranslateCoordinates(varStartPt, acUCS, acWorld,
'   If bSlope = False Then
    If (Abs(varEndPt(2) - varStartPt(2)) > 0.00001) Then
        If varEndPt(2) < varStartPt(2) Then
            tmpStart = varEndPt
            varEndPt = varStartPt
            varStartPt = tmpStart
        End If
        tmpPt(0) = varEndPt(0): tmpPt(1) = varEndPt(1): tmpPt(
        Set objLine = MS.AddLine(varStartPt, tmpPt)
        dSideA = objLine.Length
        dHyp = cetoolbox.GetDist(varStartPt, varEndPt)
        dCosA = dSideA / dHyp
        dAngOffXY = cetoolbox.RtoD(cetoolbox.ArcCos(dCosA))
    End If
'   End If
End With

On Error GoTo 0
```

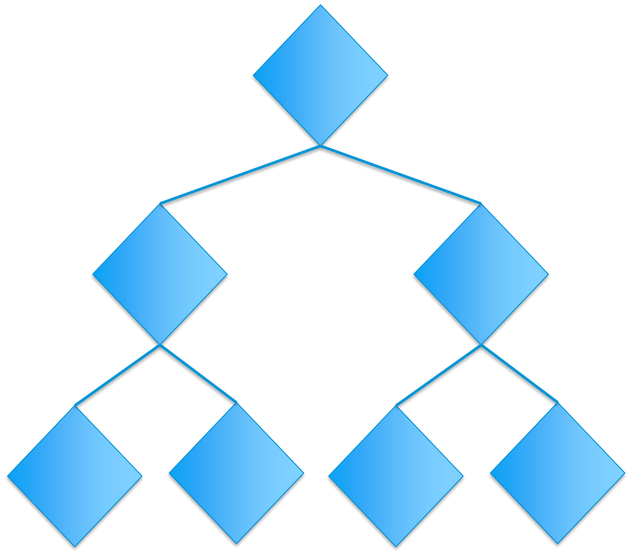


https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Functions - Avoid Nesting Conditionals

```
else
{
    if (lookupItems != null)
    {
        object match = null;
        match = ((IList<LookupItem>)lookupItems).Where(x => x.DisplayName == inputValue).FirstOrDefault();
        if (match == null)
        {
            match = ((IList<DataDictionary>)lookupItems).Where(x => x.Name == inputValue).FirstOrDefault();
            if (match == null)
            {
                MessageBoxService.ShowMessage("Invalid input. You must enter a valid item on the list.");
                return;
            }
        }

        inputValue = ((DomainObject)match).ID.ToString();
    }
    Type t = focusedColumn.FieldType;
    string typeName = ReflectionTools.GetFriendlyTypeName(t);
    foreach (var item in items)
    {
        switch (typeName)
        {
            case "System.DateTime":
            case "System.Nullable<System.DateTime>":
                DateTime dateVal;
                if (DateTime.TryParse(inputValue, out dateVal))
                    ReflectionTools.SetValue(item, focusedColumn.FieldName, dateVal);
                break;
```

Yup. Still A Pig.



Functions – Command or Query (not both)

```
public bool SetUserName(string userName)
{
    UserName = userName;
    return true;
}

public void SetUserName(string userName)
{
    UserName = userName;
}

public bool HasUserName()
{
    return !string.IsNullOrEmpty(UserName);
}
```

Command: Does something

Query: Returns something

Functions – Decomplicate Conditionals

```
public bool DoStringsMatch(string source, string target)
{
    if (source == target)
    {
        return true;
    }
    else
    {
        return false;
    }
}

public bool DoStringsMatch(string source, string target)
{
    return source == target;
}
```

Functions - Avoid Unnecessary Complexity

```
if(someCondition == false)
```

```
if(someCondition != false)
```

```
if(!someCondition)
```

```
if(!someCondition == false)
```

Demo 1: CleanCodeLayer

- **Single Responsibility Principle**
  - Do one thing and do it well
- **Interface Segregation Principle**
  - Keep interfaces small and specific to the problem
- **Dependency Inversion Principle**
  - Code to Abstractions, not Concrete Implementations
  - Dependencies should come from *outside* the class

Demo 1: Original Form with too many Responsibilities

1. Display list of layers for user to select.

```
public partial class LayerSelector : Form
{
public LayerSelector()
{
    InitializeComponent();
}

private void LayerSelector_Load(object sender, EventArgs e)
{
    var layers = new List<string>();
    var doc = AcadApplication.DocumentManager.MdiActiveDocument;
    var db = doc.Database;

    using (var transaction = doc.TransactionManager.StartTransaction()
    {
        var layerTable = (LayerTable)transaction.GetObject(db.LayerT
OpenMode.ForRead);
        foreach (var layerId in layerTable)
        {
            var layer = (LayerTableRecord)transaction.GetObject(layerId,
OpenMode.ForRead);
            layers.Add(layer.Name);
        }
    }

    Layers.DataSource = layers;
}
```

Coding to concrete implementation (Autocad database)

2. Extract list of layers from active drawing.

```
private void Ok_Click(object sender, EventArgs e)
{
    string layerName = (string)Layers.SelectedItem;
    if (!string.IsNullOrEmpty(layerName))
    {
        var doc = AcadApplication.DocumentManager.MdiActiveDocument;
        var db = doc.Database;
        using (var transaction = doc.TransactionManager.StartTransaction())
        {
            var layerTable =
            ransaction.GetObject(db.LayerTableId, OpenMode.ForRead);
            if (layerTable.Has(layerName))
            {
                var layer = transaction.GetObject(layerTable[layerName],
OpenMode.ForRead);
                db.Clayer = layer.ObjectId;

            }
            transaction.Commit();
        }
    }
}
}
```

3. Set the current layer to selected layer

Demo 1: Program to Abstraction (interface)

```
public interface ILayerProvider
{
    IList<string> GetLayers();
}
```

Display list of layers for user to select.

```
public partial class LayerSelectorClean : Form
{
    ILayerProvider _layerProvider;
    public LayerSelectorClean(ILayerProvider layerProvider)
    {
        InitializeComponent();
        _layerProvider = layerProvider;
    }

    private void LayerSelectorClean_Load(object sender, EventArgs e)
    {
        Layers.DataSource = _layerProvider.GetLayers();
    }

    public string SelectedLayer => (string)Layers.SelectedItem;
}
```

Demo 1: Implement ILayerProvider

```
class InMemoryLayerProvider : ILayerProvider
{
    public IList<string> GetLayers()
    {
        return new List<string>
        {
            "C-ALGN-ROAD-BRNG",
            "C-ALGN-ROAD-CURV",
            "C-ALGN-SSWR-GEOM",
            "C-ALGN-SSWR-TEXT"
        };
    }
}
```

Create a list of layers.

Demo 1: Implement ILayerProvider

```
class ActiveDrawingLayerProvider : ILayerProvider
{
    public IList<string> GetLayers()
    {
        var doc = AcadApplication.DocumentManager.MdiActiveDocument;
        var db = doc.Database;

        var layers = new List<string>();
        using (var transaction = doc.TransactionManager.StartTransaction())
        {
            var layerTable = (LayerTable)transaction
                .GetObject(db.LayerTableId, OpenMode.ForRead);

            foreach (var layerId in layerTable)
            {
                var layer = (LayerTableRecord)transaction
                    .GetObject(layerId, OpenMode.ForRead);
                layers.Add(layer.Name);
            }
        }

        return layers;
    }
}
```

Extract a list of layers from the Active Drawing.

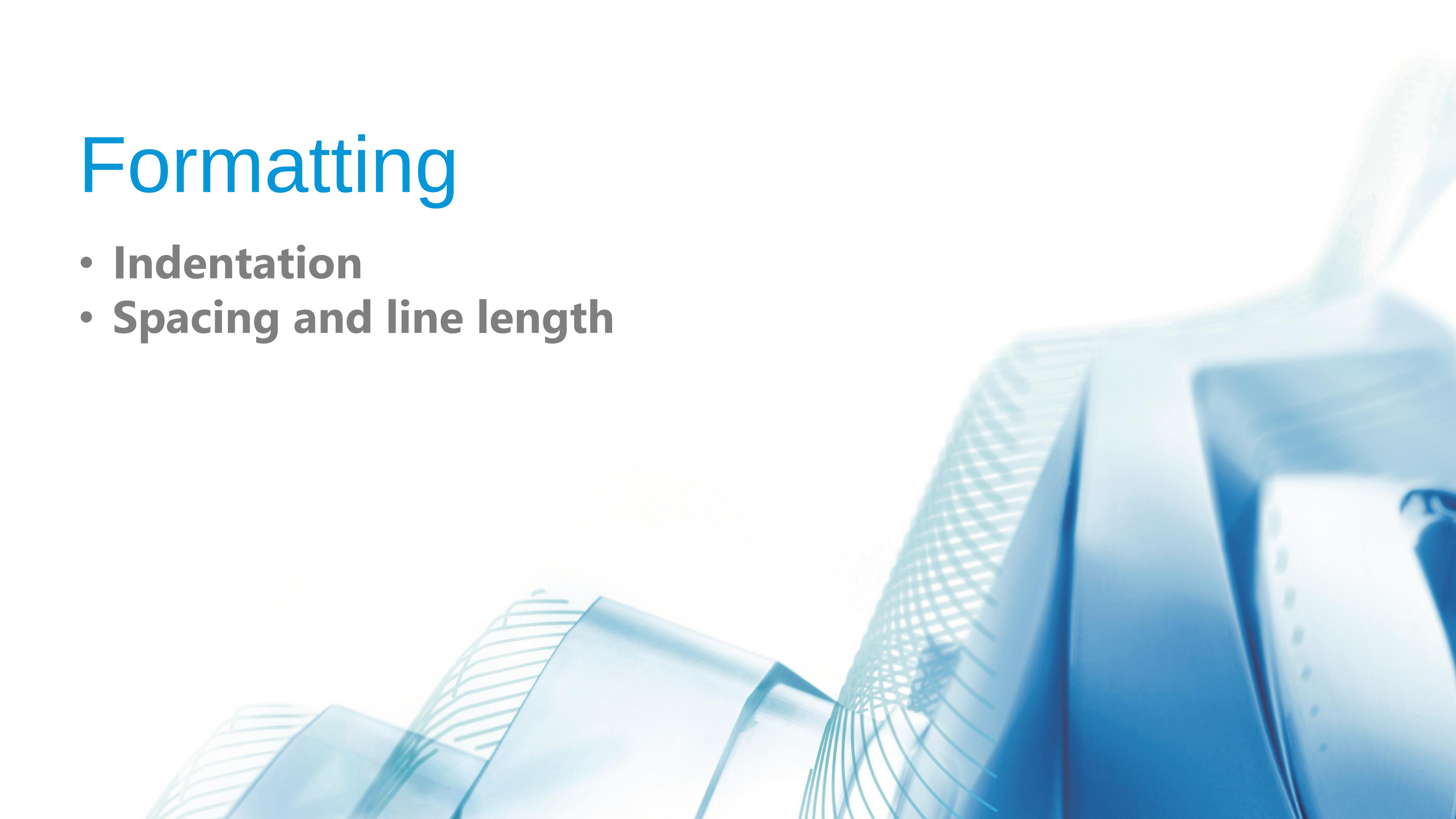
Demo 1: New Commands

```
[CommandMethod("SETLAYER2")]
public void SetLayer2()
{
    var form = new LayerSelectorClean(new InMemoryLayerProvider());
    if(form.ShowDialog() == DialogResult.OK)
    {
        SetCurrentLayer(form.SelectedLayer);
    }
}

[CommandMethod("SETLAYER3")]
public void SetLayer3()
{
    var form = new LayerSelectorClean(new ActiveDrawingLayerProvider());
    if(form.ShowDialog() == DialogResult.OK)
    {
        SetCurrentLayer(form.SelectedLayer);
    }
}
```

Sets the current layer.
(extracted from OK_Click)

Formatting

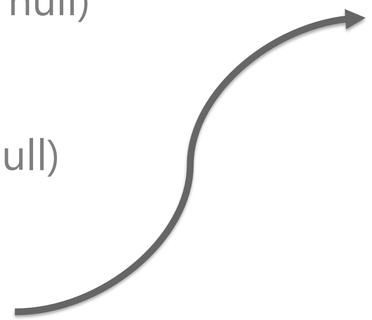
- **Indentation**
- **Spacing and line length**



Formatting - Indentation

```
var rows=prompt("How many rows for your multiplication t
able?"); var cols=prompt("How many columns for your mult
iplication table?"); if(rows=="" || rows==null) rows=10; if(col
s=="" || cols==null) cols=10; createTable(rows, cols); functio
n createTable(rows, cols){var j=1; var output="<table border
='1' width='500' cellspacing='0'cellpadding='5'>"; for(i=1;i
<=rows;i++){output=output + "<tr>"; while(j<=cols){outpu
t=output + "<td>" + i*j + "</td>"; j=j+1;}output=output +
"</tr"; j=1;}output=output + "</table>"; document.write(o
utput);}
```

```
var rows = prompt("How many rows for your multiplication table?");
var cols = prompt("How many columns for your multiplication table?");

if(rows == "" || rows == null)
    rows = 10;

if(cols == "" || cols == null)
    cols = 10;

createTable(rows, cols);
```



```
function createTable(rows, cols)
{
    var output = "<table border='1' width='500' cellspacing='0' cellpadding='5'>";

    var i = 1;
    var j = 1;
    for(i = 1;i <= rows;i++)
    {
        output = output + "<tr>";
        while(j <= cols)
        {
            output = output + "<td>" + i * j + "</td>";
            j = j+1;
        }

        output = output + "</tr>";
        j = 1;
    }

    output = output + "</table>";
    document.write(output);
}
```

```
namespace CleanCode {
  class LayerService {
    public IEnumerable<string> GetLayers() {
      if(condition == true) {
        //do something
      }
      return layers;
    }
  }
}
```

```
namespace CleanCode
{
  class LayerService
  {
    public IEnumerable<string> GetLayers()
    {
      if(condition == true)
      {
        //do something
      }
      return layers;
    }
  }
}
```

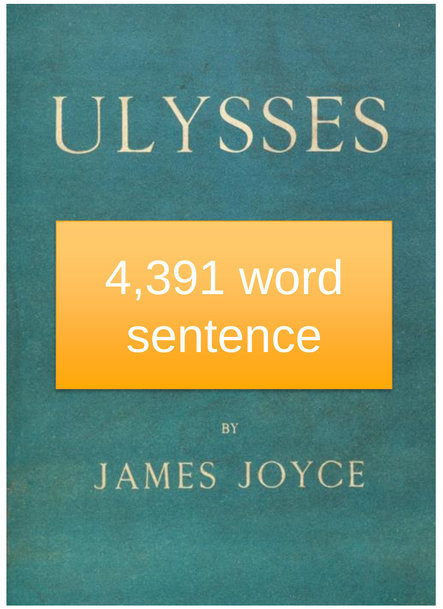
Formatting –Line Length

```
public void AvoidLongHorizontalLines()
{
    var employees = new List<Employee>();
    var emps = employees.Where(e => e.FirstName.StartsWith("B")).OrderBy(e => e.FirstName).ThenBy(e => e.LastName).Select(e => $"{e.LastName}, {e.FirstName}");
}
```

```
public void AvoidLongHorizontalLines()
{
    var employees = new List<Employee>();
    var emps = employees
        .Where(e => e.FirstName.StartsWith("B"))
        .OrderBy(e => e.FirstName)
        .ThenBy(e => e.LastName)
        .Select(e => $"{e.LastName}, {e.FirstName}");
}
```

Formatting – Spacing and Line Length

```
public void Draw()
{
    Editor editor = Application.DocumentManager.MdiActiveDocument.Editor;
    Document document = Application.DocumentManager.MdiActiveDocument;
    Database database = Application.DocumentManager.MdiActiveDocument.Database;
    PromptPointOptions startPointOptions = new PromptPointOptions("Pick start point: ");
    PromptPointResult pointResult = editor.GetPoint(startPointOptions);
    if (pointResult.Status != PromptStatus.OK) return;
    Point3d startPoint = pointResult.Value;
    PromptPointOptions endPointOptions = new PromptPointOptions("Pick end point:");
    endPointOptions.BasePoint = startPoint;
    pointResult = editor.GetPoint(endPointOptions);
    if (pointResult.Status != PromptStatus.OK) return;
    Point3d endPoint = pointResult.Value;
    using (Transaction transaction = document.TransactionManager.StartTransaction())
    {
        BlockTable blockTable = (BlockTable)transaction.GetObject(database.BlockTableId, OpenMode.ForRead);
        BlockTableRecord modelSpace = (BlockTableRecord)transaction.GetObject(blockTable[BlockTableRecord.ModelSpace], OpenMode.ForRead);
        Line line = new Line(startPoint, endPoint);
        modelSpace.AppendEntity(line);
        transaction.AddNewlyCreatedDBObject(line, true);
        transaction.Commit();
    }
}
```



```
public void Draw()
{
    Document document = Application.DocumentManager.MdiActiveDocument;
    Editor editor = doc.Editor;
    Database database = doc.Database;

    PromptPointOptions startPointOptions = new PromptPointOptions("Pick start point: ");
    PromptPointResult pointResult = editor.GetPoint(startPointOptions);
    if (pointResult.Status != PromptStatus.OK)
          return;

    Point3d startPoint = pointResult.Value;
    PromptPointOptions endPointOptions = new PromptPointOptions("Pick end point:");
    endPointOptions.BasePoint = startPoint;
    pointResult = editor.GetPoint(endPointOptions);
    if (pointResult.Status != PromptStatus.OK)
          return;

    Point3d endPoint = pointResult.Value;
    using (Transaction transaction = document.TransactionManager.StartTransaction())
    {
        BlockTable blockTable = (BlockTable)transaction.GetObject(database.BlockTableId, OpenMode.ForRead);
        BlockTableRecord modelSpace = (BlockTableRecord)transaction.GetObject(blockTable[BlockTableRecord.ModelSpace], OpenMode.ForRead);

        Line line = new Line(startPoint, endPoint);
        modelSpace.AppendEntity(line);

        transaction.AddNewlyCreatedDBObject(line, true);
        transaction.Commit();
    }
}
```

Comments

- **Need to be maintained**
- **Explain sections of code (refactor and rename?)**
- **Versioning information**
- **Zombie code**
- **TODO** and **Regions**

https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Comments – Need to be Maintained

```
(setvar "osmode" 512) ;set near on
(setq "osmode" 29) ;set end+center+node+quadrant on
```

```
/// <summary>
/// Represents a grid with rows and columns.
/// </summary>
/// <param name="startPoint"></param>
/// <param name="numRows"></param>
/// <param name="numCols"></param>
public Grid(Point3d startPoint, int numRows, int numCols, double spacing)
{
    //do something
}
```

Comments – Explanation (refactor and rename?)

```
//only do this if we're on the home page of the site
if (window.location.pathname == "/") {
  //do something
}


function isHomePage(){
  return window.location.pathname == "/";
}

if (isHomePage()) {
    //do something
}
```

Comments – Versioning Information

```
//2019-10-09 - Modified by Ben
//v1.0 First version!
//v1.1 Added comments
```

1119	brand	Friday, June 01, 2018 10:07:11	Bug fixes.
1118	brand	Thursday, May 31, 2018 12:54:26	Lots of enhancements to Phone and Contact per request by Darryl.
1117	brand	Tuesday, May 29, 2018 16:24:53	Added SortableEmployeePhoneList report.
1116	brand	Thursday, May 24, 2018 15:52:37	Added type selection for Action Items.
1115	brand	Tuesday, May 22, 2018 15:11:59	Budget implemented and documentation added.
1114	brand	Monday, May 21, 2018 16:37:15	Update to DevExpress 18.1.3.
1113	brand	Thursday, May 17, 2018 07:11:26	Merged branch 2018_05_03.
1099	brand	Wednesday, May 02, 2018 17:19:07	Reorganization into Onion Architecture--I hope.
1098	brand	Wednesday, May 02, 2018 16:06:04	Removed unused
1097	brand	Tuesday, May 01, 2018 11:09:19	Updated tag le

Source Control history

653	brand	32	ShapeDTO copeShape = shapesToCope.FirstOrDefault();
653	brand	33	if (copeShape != null)
653	brand	34	{
646	gboldt	35	foreach (ShapeDTO cutterShape in cutterShapes)
646	gboldt	36	{
653	brand	37	Point3dCollection pts = JIS_SteelWorx.Cope.Intersection.FindIntersectingPoints(copeShape, cutterShape);
646	gboldt	38	if (pts.Count > 0)
646	gboldt	39	{
653	brand	40	breakList = BreakShapeAtIntersection(copeShape, pts[
646	gboldt	41	shapesToCope.Concat(breakList);
646	gboldt	42	pts = null;
646	gboldt	43	found = true;

Source Control “Blame”

Comments – Zombie Code

```
391   brand   117         }

                          public virtual ProjectTask Task { get; set; }
                          public int? Task_ID { get; set; }
                          public string Phase { get; set; }

1339  brand   12          //public bool IsArchived
1339  brand   124         //{
1339  brand   125         //    get
1339  brand   126         //    {
1339  brand   127         //        if (Revisions.Count == 0)
1339  brand   128         //            return false;
1339  brand   129
1339  brand   130         //        return Revisions.OrderByDes
1339  brand   131         //            .First()
1339  brand   132         //            .IsArchived;
1339  brand   133         //    }
1339  brand   134         //}
```

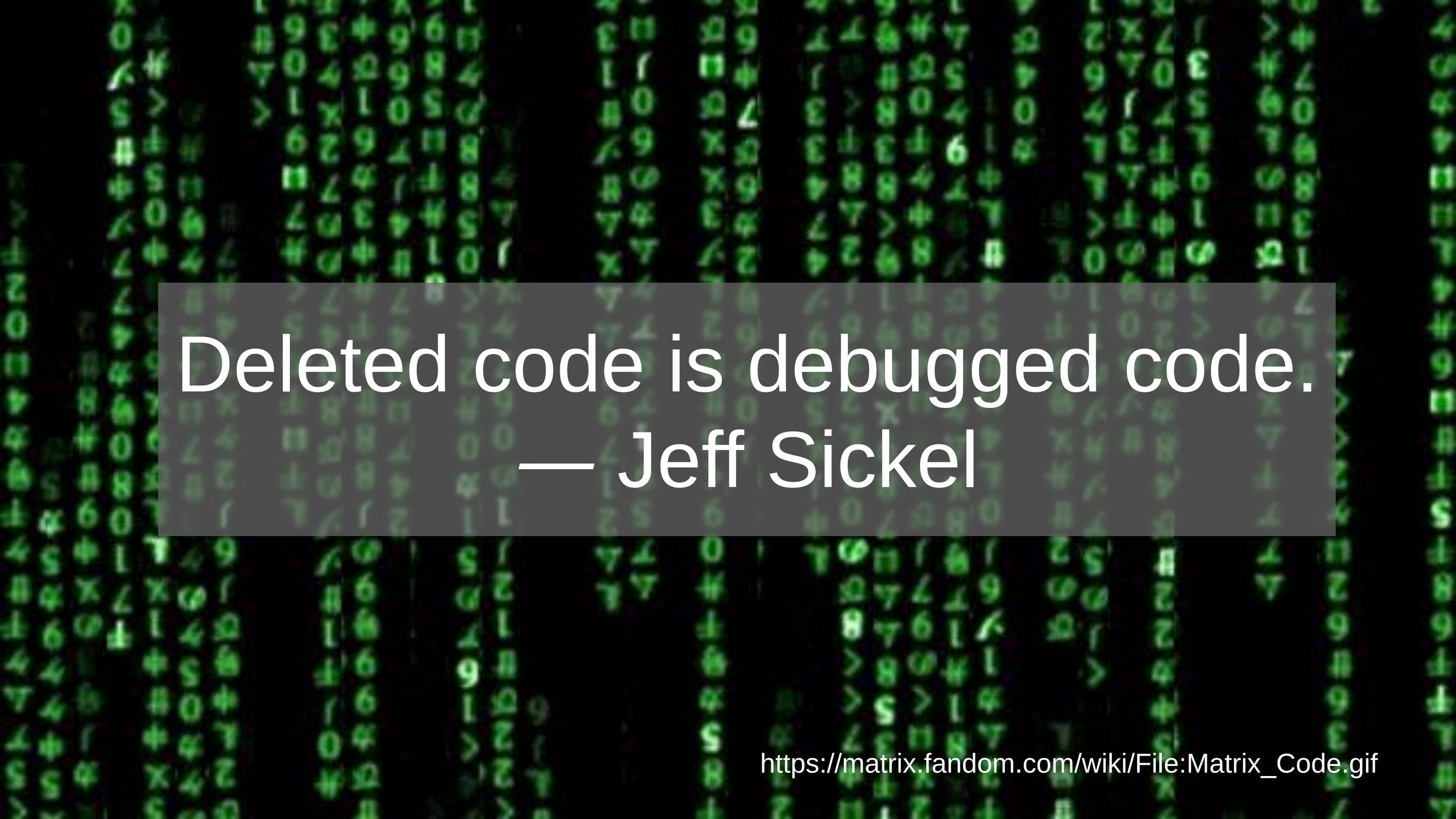
6/12/2019 16:26:34

This code is still in my project in Oct 2019!!

And was commented out in June 2019

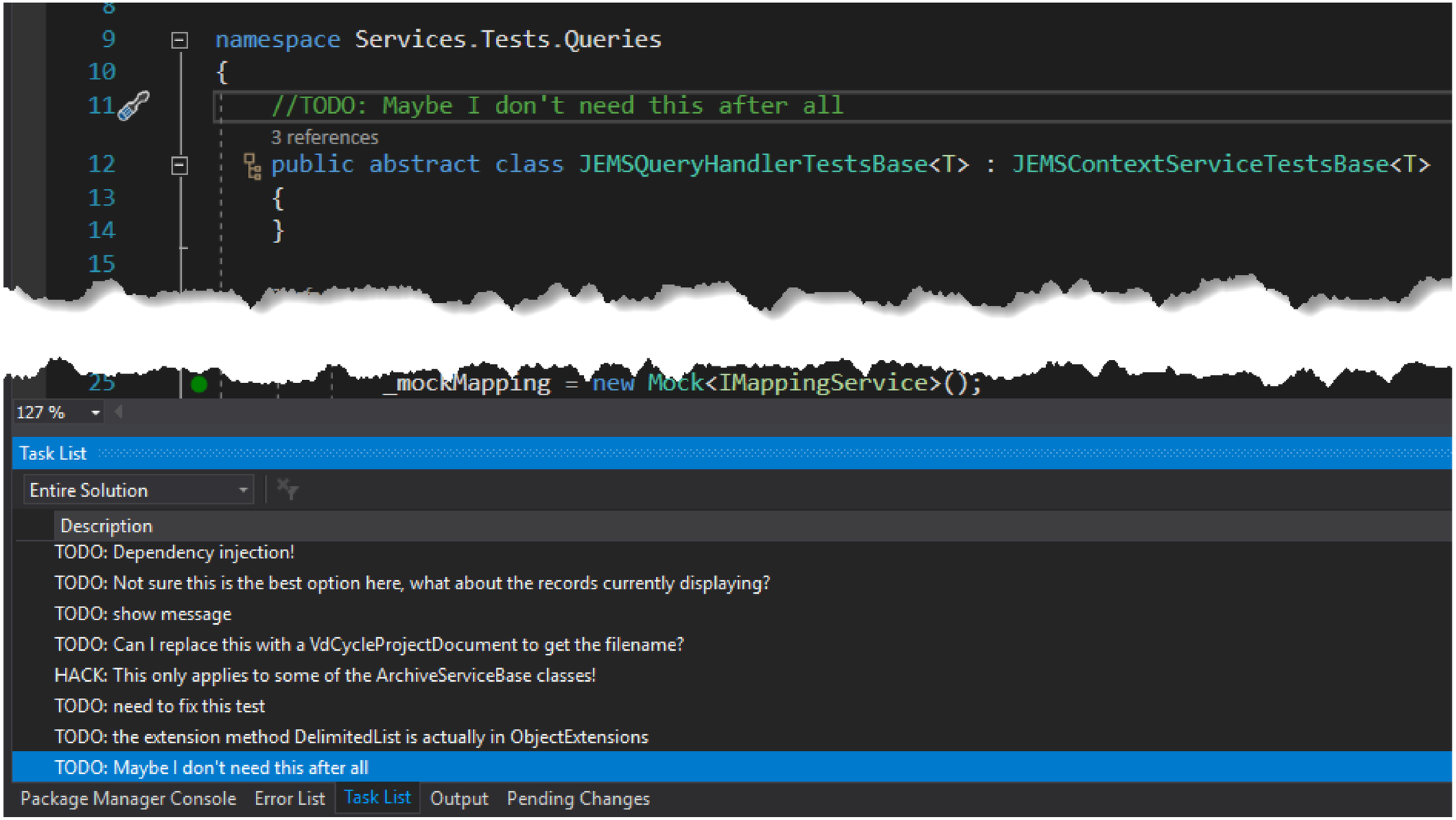
Use Source Control

Code



https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Comments – TODOs often become clutter



Comments – Regions (what are you hiding?)

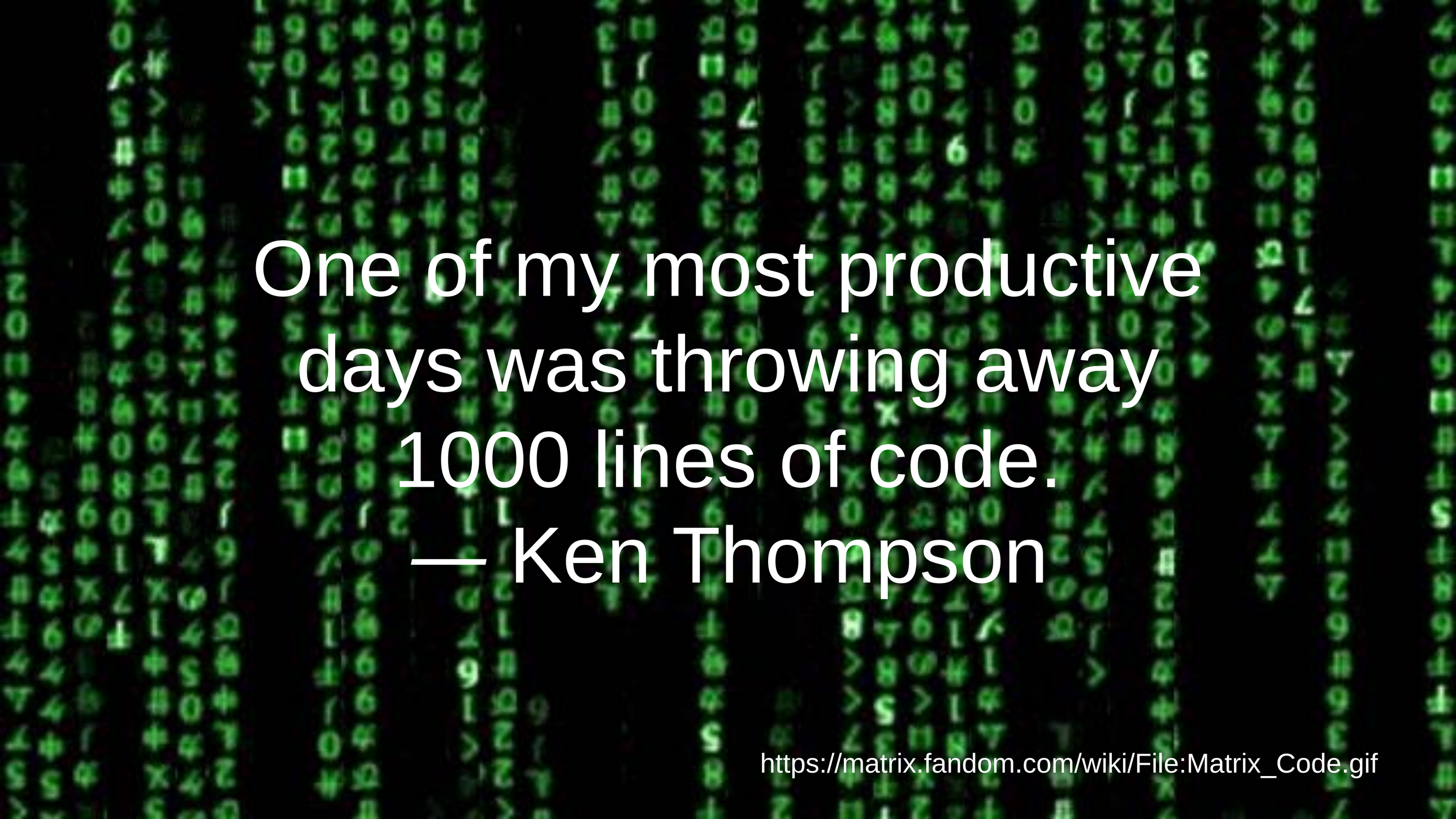
Fields

Zombie code:
Died 02/28/2019

```
#region Fields
//private bool _ShowLoadingStatus = true;
private readonly PermissionProvider _permissionProvider;
private string _RibbonStyle;
private ObservableCollection<GalleryItemGroup> _SafetyDocumentGroup;
private bool _IsSearchHidden = true;
private ObservableCollection<GalleryItemGroup> _HRDocumentGroup;
private ObservableCollection<GalleryItemGroup> _DeliverableDocumentGroup;
private ObservableCollection<GalleryItemGroup> _EquipmentDocumentGroup;
private ObservableCollection<GalleryItemGroup> _AutomationDocumentGroup;
private ObservableCollection<GalleryItemGroup> _ProjectDocumentGroup;
private Project _ActiveProject;
private ObservableCollection<Project> _Projects;
private readonly IGalleryDocumentService _galleryDocumentService;
readonly Dictionary<string, bool> _permissions;
private readonly IProjectRepository _projectService;
private readonly Dictionary<string, IDocument> documents = new Dictionary<string, IDocument>();
private DelegateCommand<DevExpress.Xpf.Ribbon.DropDownGalleryEventArgs> _dropDownGalleryInit;
private bool? _hasViewInvoicePermission;
private ErrorResultsViewModel _errorResults;
private DelegateCommand<string> showDocumentCommand;
/// <summary>
/// zBenTest project (default)
/// </summary>
private const int DEFAULT_PROJECT_ID = 883;
#endregion
```

Too many responsibilities?

Is this comment helping?
Better name for field?



https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Unit Tests

- **Organization: Arrange, Act, Assert**
- **Keep tests clean**
  - readability!
  - Tests are part of our code, too
- **Single concept per test**
- **One assertion per test**
- **F.I.R.S.T.:**
  - Fast
  - Independent
  - Repeatable
  - Self-validating
  - Thorough/Timely

Unit Tests – Test Driven Design (TDD)

Three Laws of TDD => Thou Shalt NOT:

1. Write production code until you have written a failing unit test.
2. Write more of a unit test than is sufficient to fail, and not compiling is failing.
3. Write more production code than is sufficient to pass the currently failing test.

--From *Clean Code: A Handbook of Agile Software Craftsmanship, Robert C. Martin*

TDD Tutorial:

https://blog.coryfoy.com/2006/08/tdd-bowling-game-part-1/
https://blog.coryfoy.com/2006/08/tdd-bowling-game-part-2/

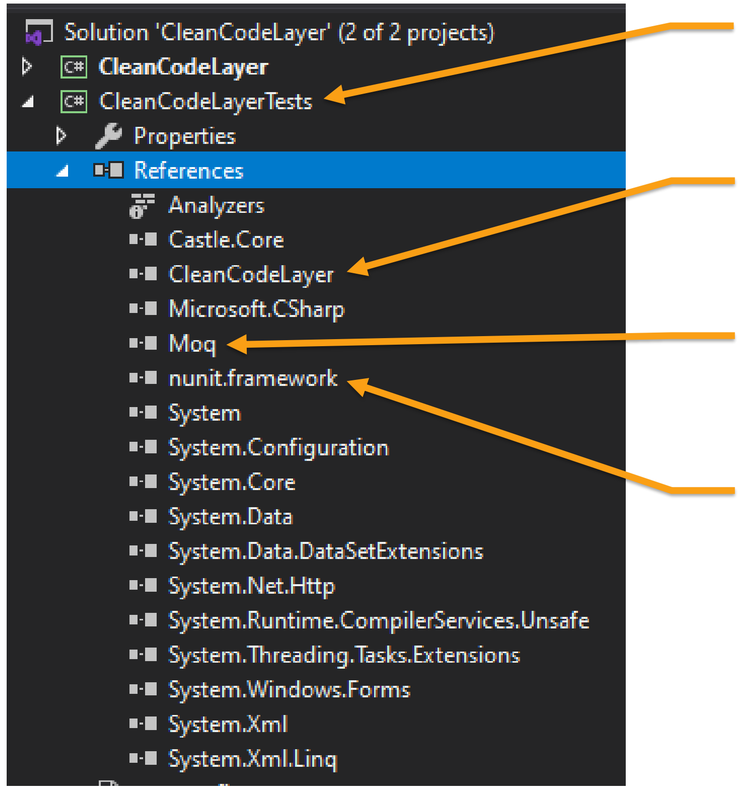
Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.
— Brian W. Kernighan

https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Demo 2: Unit Testing CleanCodeLayer

- **Unit Testing**
  - Nunit
  - Moq
- **Refactoring**
  - Often leads to more flexible code

Demo 2: Adding a Project for Unit Tests



Class Library (.NET Framework)

Project Reference to CleanCodeLayer

NuGet Package: Moq

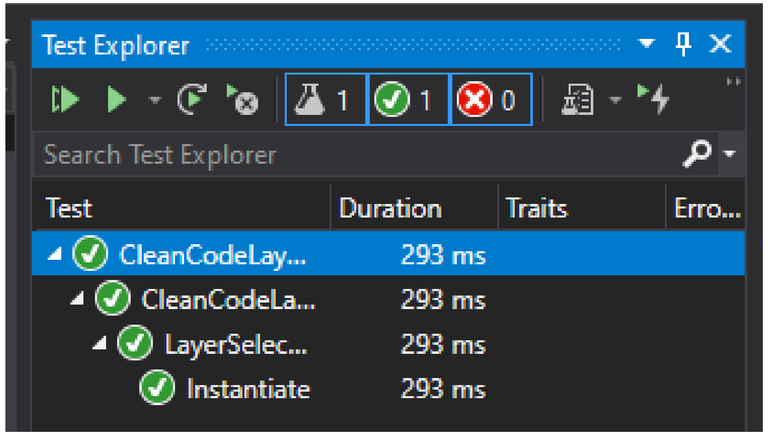
NuGet Packages: NUnit and NUnit3TestAdapter

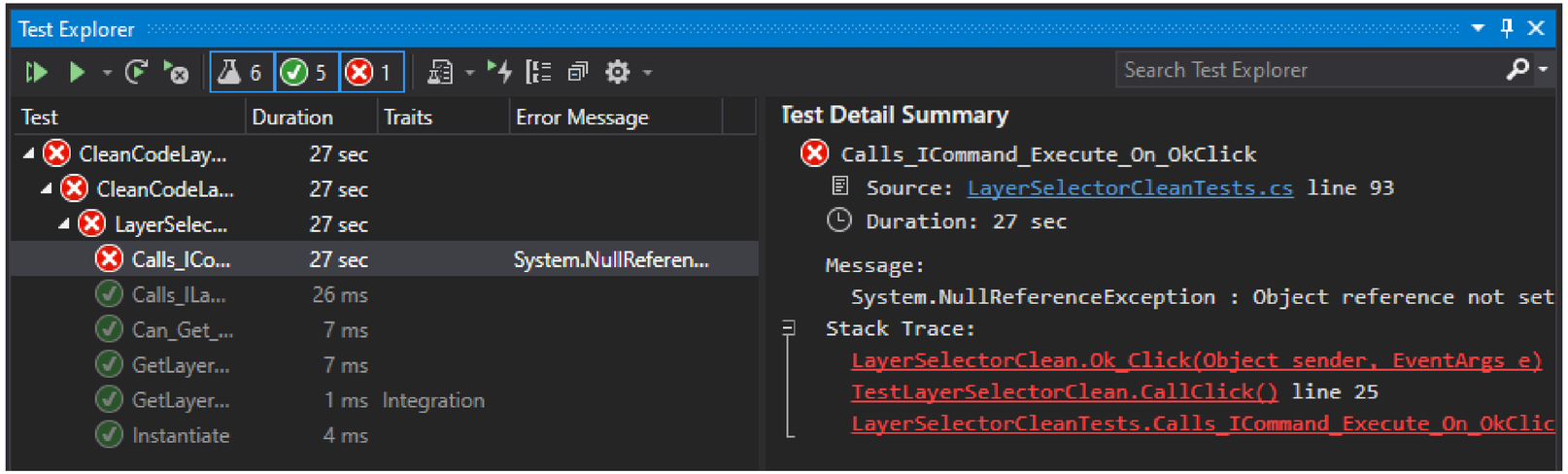
Demo 2: Test 1 - Instantiation

```
using CleanCodeLayer;
using Moq;
using NUnit.Framework;
using System;
using System.Collections.Generic;
using System.Linq;

namespace CleanCodeLayerTests
{
    [TestFixture]
    public class LayerSelectorCleanTests
    {
        [Test]
        public void Instantiate()
        {
            //Arrange
            ILayerProvider provider = new InMemoryLayerProvider();
            //Act
            var sut = new LayerSelectorClean(provider);
            //Assert
            Assert.IsNotNull(sut);
        }
    }
}
```

Required using statements

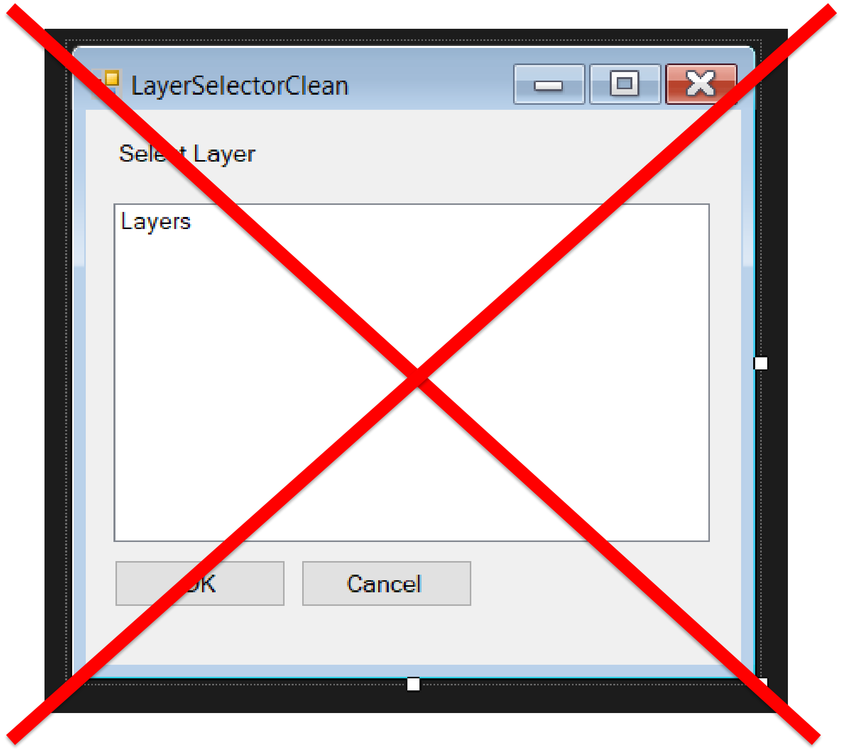
Demo 2: Using Test Runner





Demo 2: Interaction with ILayerProvider?

```
private void LayerSelectorClean_Load(object sender, EventArgs e)
{
    Layers.DataSource = _provider.GetLayers();
}
```



Demo 2: Creating a “shim” class to aid testing

```
class ShimLayerSelectorClean : LayerSelectorClean
{
    public ShimLayerSelectorClean(ILayerProvider provider) : base(provider)
    { }

    public void CallLoad()
    {
        base.OnLoad(EventArgs.Empty);
    }

    public void CallClick()
    {
        Ok_Click(Ok, EventArgs.Empty);
    }

    public ListBox LayersAccessor => base.Layers;
}
```

LayerSelectorClean:

- Change Ok_Click to *protected*
- Change Layers listbox to *protected*

Demo 2: Test 2 – Verifying interaction with mocked dependencies

```
[Test]
public void Calls_ILayerProvider_GetLayers()
{
    //Arrange
    Mock<ILayerProvider> mockProvider = new Mock<ILayerProvider>();
    var sut = new ShimLayerSelectorClean(mockProvider.Object);
    //Act
    sut.CallLoad();

    //Assert
    mockProvider.Verify(x => x.GetLayers(), Times.Once);
}
```

Demo 2: Using a SetUp method

```
Mock<ILayerProvider> _mockProvider;
ShimLayerSelectorClean _sut;
[SetUp]
public void Setup()
{
    _mockProvider = new Mock<ILayerProvider>();

    var data = new List<string> { "A", "B", "C" };
    _mockProvider.Setup(x => x.GetLayers()).Returns(data);

    _sut = new ShimLayerSelectorClean(_mockProvider.Object);
}
```

Demo 2: Test 3 – Testing that Layers are loaded

```
[Test]
public void GetLayers_From_ILayerProvider()
{
    //Arrange
    var expected = 3;
    //Act
    _sut.CallLoad();
    //Assert
    var actual = _sut.LayersAccessor.Items.Count;
    Assert.That(actual, Is.EqualTo(expected));
}
```

Demo 2: Test 4 – Get Selected Layer Name

```
//In LayerSelectorClean
public string SelectedLayer => (string)Layers.SelectedItem;
public IEnumerable<string> AllLayers => Layers.Items.Cast<string>();



//In LayerSelectorCleanTests
[Test]
public void Can_Get_Layer_Name_From_SelectedLayer()
{
    var expected = "B";
    _sut.CallLoad();

    _sut.LayersAccessor.SelectedIndex = 1;
    var actual = _sut.SelectedLayer;

    Assert.That(actual, Is.EqualTo(expected));
}
```

Demo 2: Test 5 – Integration Testing With “Real” Provider

```
[Test]
[Category("Integration")]
public void GetLayers_From_InMemoryProvider()
{
    var expected = 4;
    var provider = new InMemoryLayerProvider();
    var sut = new ShimLayerSelectorClean(provider);
    sut.CallLoad();

    var actual = sut.LayersAccessor;

    Assert.That(actual.Items.Count, Is.EqualTo(expected));
    Assert.That(actual.Items[0], Is.EqualTo("C-ALGN-ROAD-BRNG"));
}
```

Demo 2: Adding new interfaces to support injecting commands

```
public interface ILayerSelectorCommand
{
    Func<ILayerSelector, object> GetProperty { get; set; }
    void Execute(object parameter);
}
```

```
public interface ILayerSelector
{
    string SelectedLayer { get; }
    IEnumerable<string> AllLayers { get; }
}
```

Demo 2: Updating LayerSelectorClean

```
public partial class LayerSelectorClean : Form, ILayerSelector
{
    ILayerProvider _provider;
    ILayerSelectorCommand _command;

    public LayerSelectorClean(ILayerProvider provider, ILayerSelectorCommand command)
    {
        InitializeComponent();
        _provider = provider;
        _command = command;
    }

    protected void Ok_Click(object sender, EventArgs e)
    {
        var parameter = _command.GetProperty(this);
        _command.Execute(parameter);
    }
```

Demo 2: Update ShimLayerSelectorClean and Unit Tests

```
public ShimLayerSelectorClean(ILayerProvider provider, ILayerSelectorCommand command)
    : base(provider, command)


private Mock<ILayerProvider> _mockProvider;
private Mock<ILayerSelectorCommand> _mockCommand;
private ShimLayerSelectorClean _sut;
[SetUp]
public void Setup()
{
    ...
    _mockCommand = new Mock<ILayerSelectorCommand>();
    _sut = new ShimLayerSelectorClean(_mockProvider.Object, _mockCommand.Object);
}
```

Demo 2: Test 6 – Verifying OK click executes command

```
[Test]
public void Calls_ICommand_Execute_On_OkClick()
{
    _sut.CallClick();

    _mockCommand.Verify(x => x.Execute(It.IsAny<string>()), Times.Once);
}
```

Demo 2: Updating Setup method to setup mocked command

```
[SetUp]
public void Setup()
{
    ...
    _mockCommand = new Mock<ILayerSelectorCommand>();
    _mockCommand.Setup(x => x.GetProperty).Returns(x => x.SelectedLayer);
    _sut = new ShimLayerSelectorClean(_mockProvider.Object, _mockCommand.Object);
}
```

Demo 2: CurrentLayerCommand

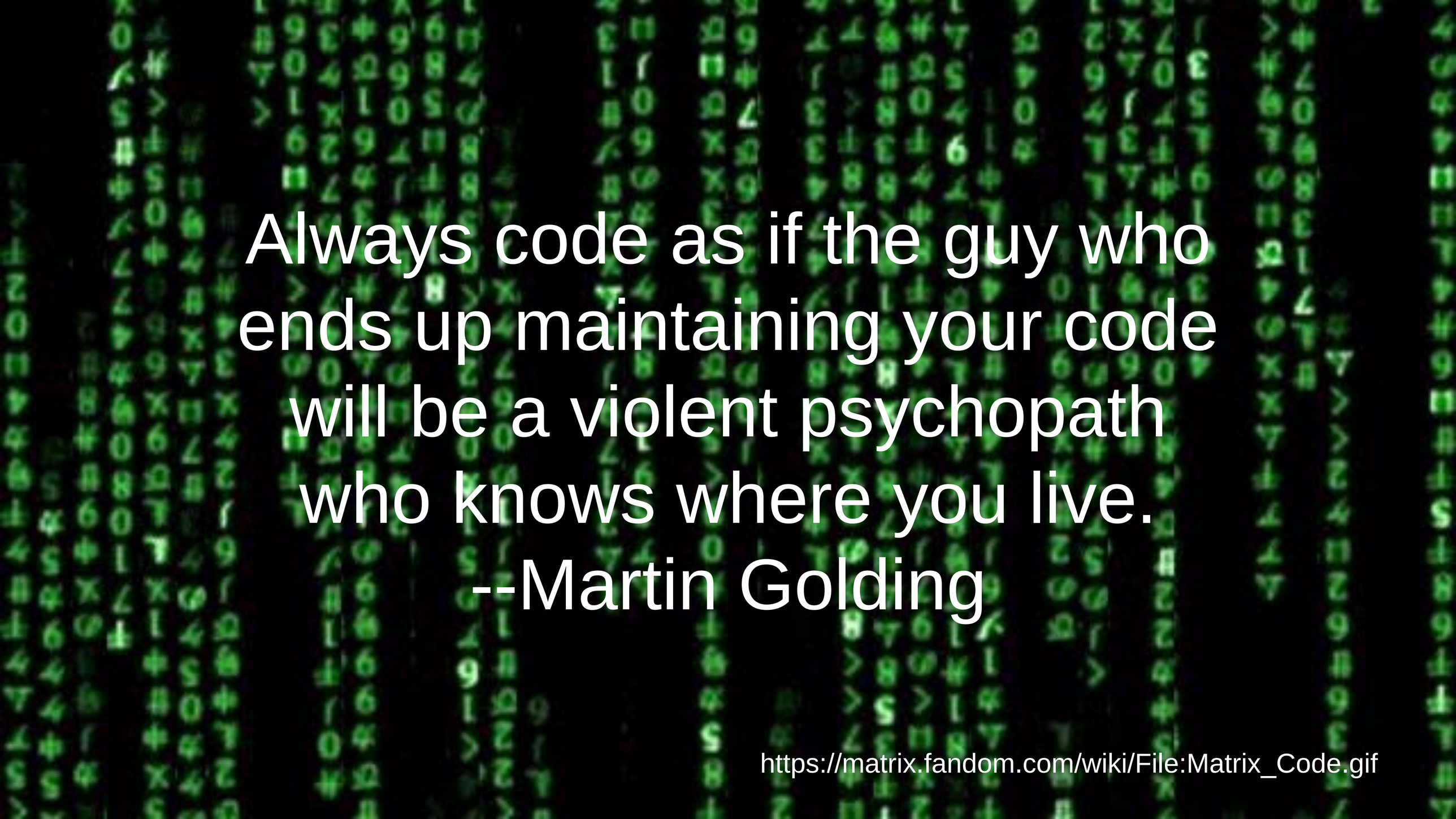
```
public class CurrentLayerCommand : ILayerSelectorCommand
{
    public CurrentLayerCommand()
    {
        GetProperty = x => x.SelectedLayer;
    }

    public Func<ILayerSelector, object> GetProperty { get; set; }

    public void Execute(object parameter)
    {
        var layerName = (string)parameter;
        Active.UsingTransaction(transaction =>
        {
            var layerTable = (LayerTable)transaction
                .GetObject(Active.Database.LayerTableId, OpenMode.ForRead);

            LayerTableRecord layer;
            if (layerTable.Has(layerName))
            {
                layer = (LayerTableRecord)transaction
                    .GetObject(layerTable[layerName], OpenMode.ForRead);
            }
            else
            {
                layer = new LayerTableRecord { Name = layerName };

                layerTable.UpgradeOpen();
                layerTable.Add(layer);
                transaction.AddNewlyCreatedDBObject(layer, true);
            }

            Active.Database.Clayer = layer.ObjectId;
        });
    }
}
```

Demo 2: LayersToEditorCommand

```
class LayersToEditorCommand : ILayerSelectorCommand
{
    public Func<ILayerSelector, object> GetProperty { get; set; }
    public LayersToEditorCommand()
    {
        GetProperty = x => x.AllLayers;
    }

    public void Execute(object parameter)
    {
        var layers = (IEnumerable<string>)parameter;

        foreach (var layer in layers)
        {
            Active.Editor.WriteMessage($"\n{layer}");
        }
    }
}
```

Demo 2: Payoff… “Composable” programming

```
[CommandMethod("SETLAYER2")]
public void SetLayer2()
{
    var form = new LayerSelectorClean(new InMemoryLayerProvider(),
        new CurrentLayerCommand());
    form.ShowDialog();
}

[CommandMethod("SETLAYER3")]
public void SetLayer3()
{
    var form = new LayerSelectorClean(new ActiveDrawingLayerProvider(),
        new CurrentLayerCommand());
    form.ShowDialog();
}

[CommandMethod("SETLAYER4")]
public void SetLayer4()
{
    var form = new LayerSelectorClean(new ActiveDrawingLayerProvider(),
        new LayersToEditorCommand());
    form.ShowDialog();
}
```



https://matrix.fandom.com/wiki/File:Matrix_Code.gif

Recommended Reading

- *Clean Code: A Handbook of Agile Software Craftsmanship*, Robert C. Martin
- *Working Effectively with Legacy Code*, Michael C. Feathers
- *Dependency Injection in .NET*, Mark Seeman
- *Head First Design Patterns*, Eric Freeman
- *The Art of Unit Testing: With Examples in .NET*, Roy Osherove
- *Design Patterns: Elements of Reusable Object-Oriented Software*, Erich Gamma, et al.
- *The Pragmatic Programmer: From Journeyman to Master*, Andy Hunt
- *Refactoring: Improving the Design of Existing Code*, Martin Fowler
- *Code Complete*, Steve McConnell

Contact Me

- Email: ben@leadensky.com
- Twitter: @leadensky
- LinkedIn: www.linkedin.com/in/leadensky

Previous Courses at AU:

- [2018] AS197578-L: Data Mining in AutoCAD with Data Extraction
- [2018] SD195561: Control Your Code: Introduction to Source Control for Programmers and CAD Managers
- [2018] SD226637: Clean Code: Tips for Writing Clear and Concise Code
- [2017] Data Mining, Control Your Code (similar to 2018)
- [2016] Data Mining (similar to 2018)
- [2015] SD10672: Exploring Entity Framework for AutoCAD Development – And Beyond

Pluralsight Course

- https://www.pluralsight.com/courses/autocad-extracting-data-drawings

AUTODESK®

Make anything™