

"Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live."

MARTIN GOLDING.



@riverglide



@andypalmer



@antonymarciano



@jmrtn



@pro_cessor





Ψ

MANAGERBROKERDISPATCHER
INTERFACEIMPL

<http://www.classnamer.com/>

CheckedGraphContext

Stateles

ErrorCorrectingMessageGeneratorsRecordGenerator

CLEAN CODE *

****The book!***

Use

INTENTION**REVEALING**

Names

ABSICHTS**VERMITTELNDE**

Vermeide

DESINFORMATION

Erzeuge

AUSSAGEKRÄFTIGE

Unterschiede

Verwende
SUCHBARE
Namen

Verwende
AUSSPRECHBARE
Namen

private Date
genymdhms

private Date
generationTimestamp

```
/// <summary>  
///     Gets or sets.  
///     Used for Ewiomc.  
/// </summary>  
/// <remarks>  
///     Used internally by the bl.  
/// </remarks>  
public string Vdewgvwid { get; set; }
```

WTF? Ich geh heim.

Aussage

KLASSEN**NAMEN**

kräftig

“CLASSES AND OBJECTS SHOULD HAVE NOUN OR NOUN PHRASE NAMES LIKE CUSTOMER, WIKIPAGE, ACCOUNT, AND ADDRESSPARSER. AVOID WORDS LIKE MANAGER, PROCESSOR,, DATA, OR INFO IN THE NAME OF A CLASS. A CLASS NAME SHOULD NOT BE A VERB.”

“CLASSES AND OBJECTS SHOULD HAVE NOUN OR NOUN PHRASE NAMES LIKE CUSTOMER, WIKIPAGE, ACCOUNT, AND ADDRESSPARSER. AVOID WORDS LIKE MANAGER, PROCESSOR,, DATA, OR INFO IN THE NAME OF A CLASS. A CLASS NAME SHOULD NOT BE A VERB.”

“CLASSES AND OBJECTS SHOULD HAVE
NOUN OR NOUN PHRASE NAMES LIKE
CUSTOMER, WIKIPAGE, ACCOUNT, AND
ADDRESSPARSER. AVOID WORDS LIKE
MANAGER, PROCESSOR,, DATA, OR INFO IN
THE NAME OF A CLASS. A CLASS NAME
SHOULD NOT BE A VERB.”

“CLASSES AND OBJECTS SHOULD HAVE NOUN OR NOUN PHRASE NAMES LIKE **CUSTOMER, WIKIPAGE, ACCOUNT, AND ADDRESSPARSER**. AVOID WORDS LIKE **MANAGER, PROCESSOR,, DATA, OR INFO** IN THE NAME OF A CLASS. A CLASS NAME SHOULD NOT BE A VERB.”

“CLASSES AND OBJECTS SHOULD HAVE NOUN OR NOUN PHRASE NAMES LIKE CUSTOMER, WIKIPAGE, ACCOUNT, AND ADDRESSPARSER. **AVOID** WORDS LIKE **MANAGER, PROCESSOR, DATA, OR INFO** IN THE NAME OF A CLASS. A CLASS NAME SHOULD NOT BE A VERB.”

**WAR
UM?**

Kenne

DEINEN FEIND

Und respektiere ihn.

WEASEL WORDS

"I can suck melancholy out
of a song, as a weazel
sucks eggs."

SHAKESPEARE, AS YOU LIKE IT, II. 5..

Klassifikation

TAXONOMIE

Ordnung

The image consists of three squares arranged horizontally. The first square on the left is blue and contains the word 'Hungarian'. The second square in the middle is grey and contains the words 'Party' and 'Hats' stacked vertically. The third square on the right is also grey and contains the words 'Philo' and 'sophers' stacked vertically.

Hungarian

Party
Hats

Philo
sophers

DOUBLE INT

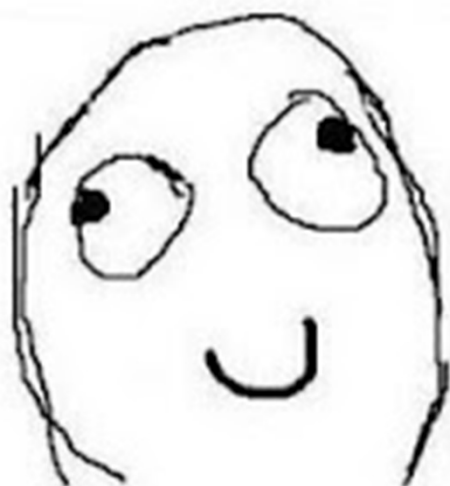
LONG STRING

pWindow

cCustomers

```
[DllImport("user32.dll")]  
static extern bool CloseWindow(IntPtr hWnd);
```

```
[DllImport("user32.dll", SetLastError=true)]
static extern IntPtr CreateWindowEx(
    WindowStylesEx dwExStyle,
    string lpClassName,
    string lpWindowName,
    WindowStyles dwStyle,
    int x,
    int y,
    int nWidth,
    int nHeight,
    IntPtr hWndParent,
    IntPtr hMenu,
    IntPtr hInstance,
    IntPtr lpParam);
```



dafuq did i just read


```
public int SumIntegersUpTo(int bound)
{
    return Enumerable.Range(1, bound).Sum();
}
```

```
private readonly ICanStartAndStop _counter;  
private readonly Wristwatch _wristwatch;  
private Brush _color;  
private double _fontSize;  
private bool _isRunning = false;  
private TimeSpan _timeLeft;
```

Hungarian

Party
Hats

Philo
sophers

```
for(int i = 0; i < customers.Count; i++) {  
    Customer theCustomer = customers[i];  
    ...  
}
```

```
foreach(var customer in customers) {  
    ...  
}
```

Hilfsmittel zur
MEHRFACHEN
Vergabe des selben Namens

MY

THE

AN

IT

Hungarian

Party
Hats

Philo
sophers

DATA

INFO

FUNCTION

PROCESS

SYSTEM

MODEL



Vanity

Details

Eitle Wichtigtuer

IMPORTANCE

Aufgeblasen!

FLEXIBLE

GENERAL

EXTENDED

SUPER

ABSTRACT

```
public abstract class AbstractTopLevelItem  
extends AbstractItem  
implements TopLevelItem {  
  
    // ...  
}
```

Die Schönheit liegt
SIMPLICITY
im Auge des Betrachters

BASIC

EASY

NEW

SPECIAL

SIMPLE



Vanity

Details

Mit Förmchen spielen...

SHAPE

Details

LIST
QUERY
ARRAY
DICTIONARY
VIEW

Benimm dich!

BEHAVIORAL

Details

DYNAMIC

LAZY

LOCAL

CONSTANT

GLOBAL

Klappts?

TEST

Details

MOCK

STUB

FAKE

DYNAMICMOCK

STRICTMOCK

MOCKS

Sind keine Stubs.

```
var stubUserRepository =  
    MockRepository.GenerateStub<IUserRepository>();
```

```
stubUserRepository.Stub(x =>  
    x.GetUserByName("ayende")).Return(theUser);
```



Roles

Patterns

Eager

PoEAA

Lazy

GoF

Charity

DDD

Übereifrig

EAGER ROLES

Tausendsassa

PROVIDER

AGENT

DISPATCHER

BROKER

MANAGER

PROZEDURALER CODE
GROSSE KLASSEN

SCHWER TESTBAR

1000+LOC

JENKINS

<https://github.com/jenkinsci/jenkins/blob/master/core/src/main/java/jenkins/model/Jenkins.java>

Faulpelze

LAZY ROLES

Nicht meine Aufgabe!

NOMINA AGENTIS

Von einem Verb

abgeleitetes Substantiv

DERIVATIONSMORPHEME

-ER

-OR

FORMATTER

TRANSFORMER

WRAPPER

MAPPER

VALIDATOR



UTILS

HELPER

“We are blind to the
World within us,
Waiting to be born”

**AT THE GATES
BLINDED BY FEAR**



Roles

Patterns

Eager

PoEAA

Lazy

GoF

Charity

DDD

Design Patterns

ENTWURFSMUSTER

Sind keine Lösung

BUILDER
FACADE
STRATEGY
DECORATOR

SINGLETON

```
var instance = ConnectionSingleton.Instance;
```

```
public static DatabaseConnectionSingleton GetInstance()
{
    return _instance
        ?? (_instance = new DatabaseConnectionSingleton());
}
```

NEW, DELETE?

**KANN DAS NICHT DIE RUNTIME
MACHEN?**

```
var instance = Only.One<DatabaseConnection>();
```

```
var instance = The.Same<DatabaseConnection>();
```

```
var instance = Unity.Resolve<DatabaseConnection>();
```



```
var instance = ObjectFucktory.GetInstance<DatabaseConnection>();
```

FACTORY

“Du willst doch nicht in einer
Pizzafabrik essen, sondern in
einem Restaurant”

**ROBERTO BEZ.
SONNTAG ABEND**

**RATHER THAN “A GENERAL REUSABLE
SOLUTION TO A COMMONLY OCCURRING
PROBLEM”, I CURRENTLY THINK OF DESIGN
PATTERNS AS A SHARED VOCABULARY
FOR DISCUSSING THE OBSERVABLE
COMMONALITIES BETWEEN TWO OR MORE
SOLUTIONS, AFTER THEY’VE EMERGED.**

RATHER THAN “A GENERAL REUSABLE
SOLUTION TO A COMMONLY OCCURRING
PROBLEM”, I CURRENTLY THINK OF DESIGN
PATTERNS AS A SHARED VOCABULARY
FOR DISCUSSING THE OBSERVABLE
COMMONALITIES BETWEEN TWO OR MORE
SOLUTIONS, AFTER THEY’VE EMERGED.

RATHER THAN “A GENERAL REUSABLE
SOLUTION TO A COMMONLY OCCURRING
PROBLEM”, **I CURRENTLY THINK** OF DESIGN
PATTERNS AS A SHARED VOCABULARY
FOR DISCUSSING THE OBSERVABLE
COMMONALITIES BETWEEN TWO OR MORE
SOLUTIONS, AFTER THEY’VE EMERGED.

RATHER THAN “A GENERAL REUSABLE
SOLUTION TO A COMMONLY OCCURRING
PROBLEM”, I CURRENTLY THINK OF DESIGN
PATTERNS AS **A SHARED VOCABULARY
FOR DISCUSSING THE OBSERVABLE
COMMONALITIES** BETWEEN TWO OR MORE
SOLUTIONS, AFTER THEY’VE EMERGED.

RATHER THAN “A GENERAL REUSABLE SOLUTION TO A COMMONLY OCCURRING PROBLEM”, I CURRENTLY THINK OF DESIGN PATTERNS AS A SHARED VOCABULARY FOR DISCUSSING THE OBSERVABLE COMMONALITIES BETWEEN TWO OR MORE SOLUTIONS, **AFTER THEY’VE EMERGED.**



<https://twitter.com/#!/jmrtn>

<http://jmrtn.com/notes/2012/02/17/design-patterns.html>

Weg Damit!

SEEK & DESTROY

Wiesel Jagd

Refactoring

RENAME CLASS

Umbenennen

“There are many powerful refactorings, but **Rename Class** is the most powerful. It changes the way people see code and lets them notice possibilities that they might not have considered before”

MICHAEL FEATHERS.

REPOSITORY

```
public class CustomerRepository
{
    IEnumerable<Customer>
    GetCustomersByYearOfBirth(DateTime yearOfBirth)
    {
    }
}
```

```
public class CustomerRepository
{
    IEnumerable<Customer>
    GetByYearOfBirth(DateTime yearOfBirth)
    {
    }
}
```

```
public class Customers
{
    IEnumerable<Customer>
    GetByYearOfBirth(DateTime yearOfBirth)
    {
    }
}
```



```
public class Customers
{
    IEnumerable<Customer>
    BornIn(DateTime yearOfBirth)
    {
    }
}
```

```
public interface ICustomerRepository
{
    IEnumerable <Customer> BornIn(DateTime yearOfBirth);
}

public class Customers : ICustomerRepository
{
    /*...*/
}
```

```
public class Customers : ICustomerRepository
{
    /*...*/
}
```

```
public interface IFindCustomers
{
    IEnumerable <Customer> BornIn(DateTime yearOfBirth);
}
```

```
public interface IFindCustomers
{
    IEnumerable <Customer> BornIn(DateTime year);
}
```

Iformatter

IFormat

Iobserver

Iobserve

INTERFACES
DEFINIEREN VERHALTEN.

WARUM NENNEN WIR SIE NICHT
SO?

I DO SOMETHING FOR YOU

SYSTEMISCHE METAPHER

KLARE NAMEN

Inspired by and using the fonts suggested at

<http://www.labnol.org/software/tutorials/advice-select-best-fonts-for-powerpoint-presentation-slides/3355/>

Health

<http://thenounproject.com/noun/first-aid/#icon-No2208>

Andy Palmer on Singletons

<http://andypalmer.com/2008/05/singletons/>



Blue
Rgb,
0,183,255



Orange,
Rgb,
255,116,0



Green,
Rgb
0,219,0



Magenta,
#FF00FF



Light Gray,
Rgb,
191,191,191

Dark Gray,
Rgb,
64,64,64