

Objektorientierte Programmierung in LotusScript

von Bernd Hort, 28.03.2022

DominoCamp 2022





Bernd Hort

- Diplom Informatiker
- Notes Entwickler seit Version 3.3
- „Gib dem Anwender das richtige Werkzeug die Aufgabe zu erledigen“

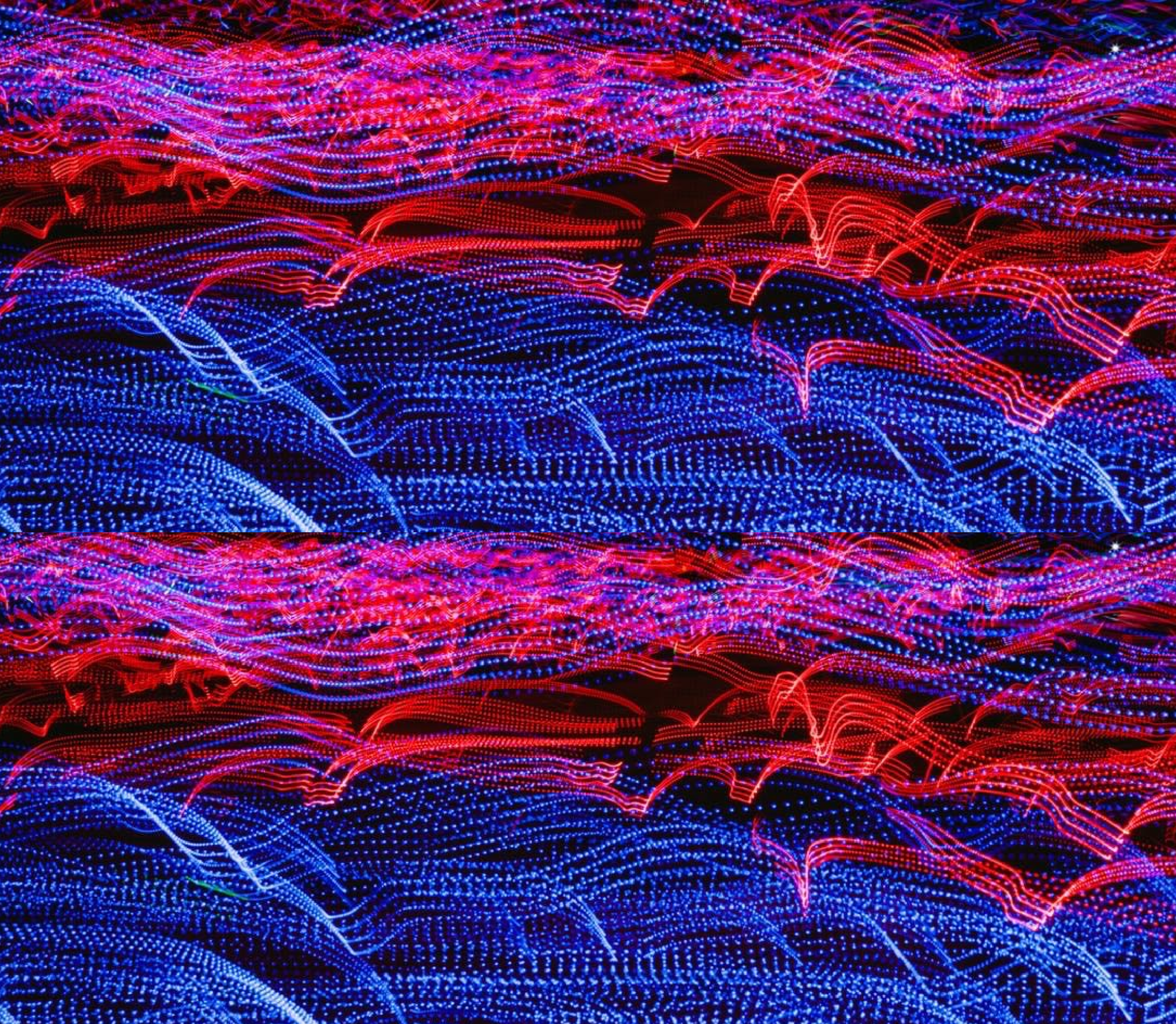
Objektorientierte Programmierung

- Einführung
- Motivation
- Syntax und Definition
- Code-Beispiele
- Fragen und Antworten



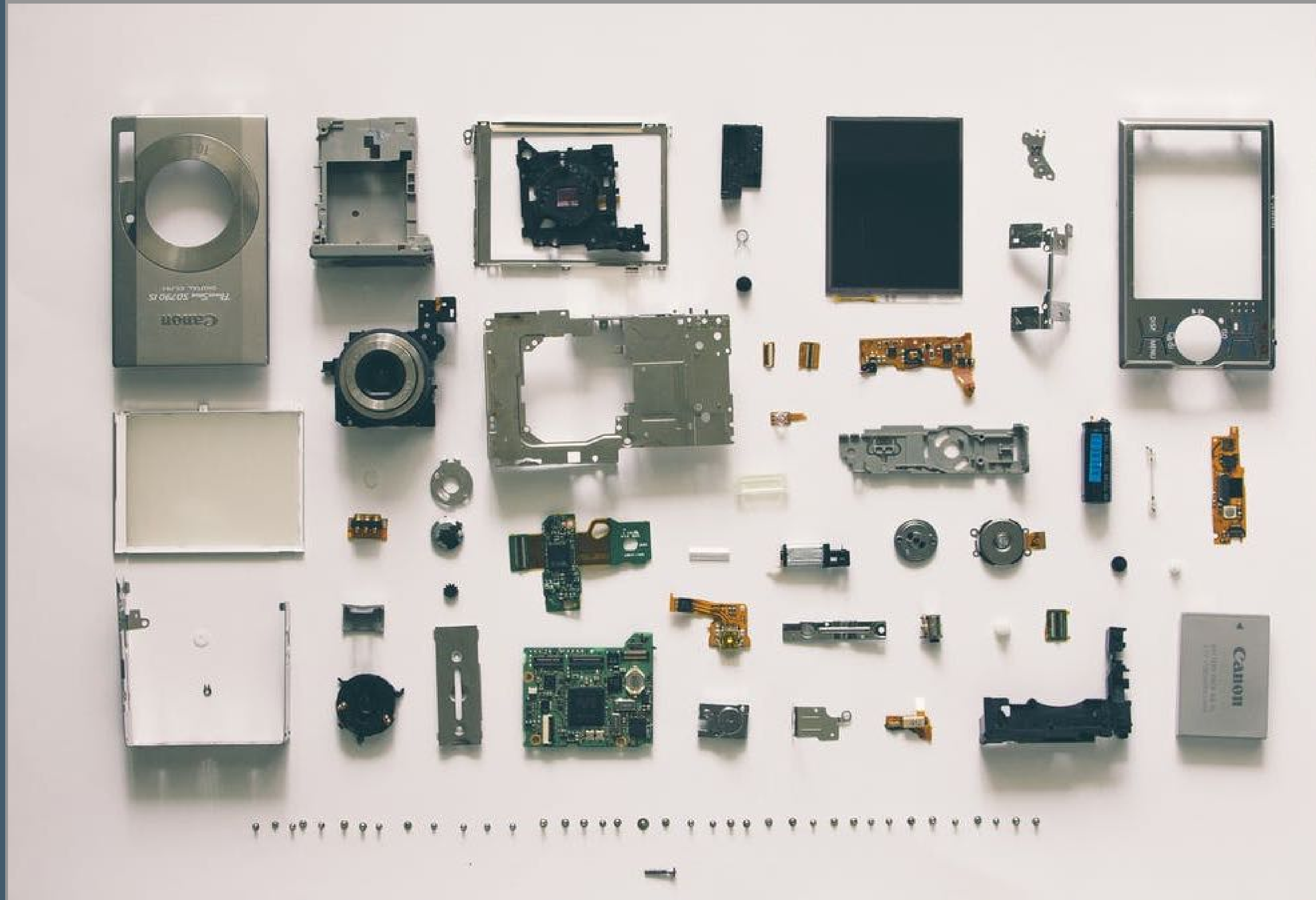
**Warum
objekt-
orientiert
programm-
ieren?**





**Höhere
Abstraktions-
ebene**

Umgang mit Komplexität



```
.removeClass("active"),a.fn.scrollspy=d,this},a(window).on("load",function(a){"use strict";function b(b){return this.each(function(){var d=a(this),e=d.data("target"),f=d.attr("href"),g=d.replace(/.*(?=#[^\s]+$)/,"").replace(/#/,""),h=d.attr("data-target");if(h&&h!=e&&f==g){d.attr("data-target",e);d.attr("data-target",e)}},a.proxy(this.checkPosition,this)).on("click.bs.affix.data-api",a.proxy(this.affix,this));c.VERSION="3.3.7",c.RESET="affix",c.prototype.getPinnedOffset=function(a,b){var e=this.$target.scrollTop(),f=this.$element.offset().top,g=this.$element.offset().bottom,h=a-d&&"bottom",i=c&&e<=c?"top":null!=d&&i+j>=a-d&&"bottom";c.prototype.getPinnedOffset=furthEventLoop=function(){setTimeout(a.proxy(this.checkPosition,this),1))nt.height(),d=this.options.offset,e=d.top,f=d.bottom,e&&(e=d.top(this.$element)),"top" ""nt.css("top" ""
```

Bessere
Lesbarkeit
des Codes

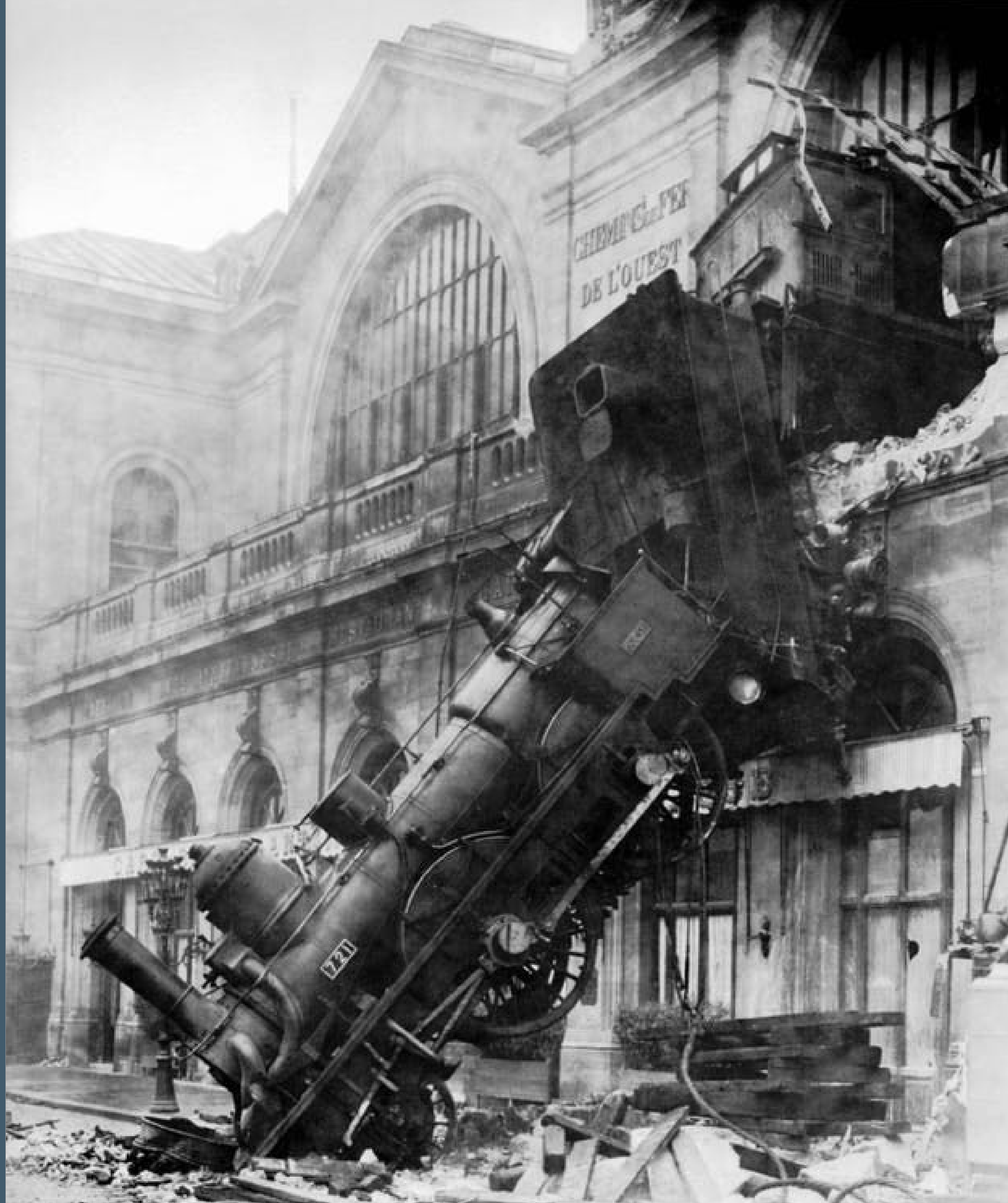
Wieder-
verwend-
barkeit





Erweiterbar
keit und
Flexibilität

Vermeidung von Fehlern





Beschleunigung der Entwicklung

Objektorientierte Programmierung

- Einführung
- Motivation
- Syntax und Definition
- Code-Beispiele
- Fragen und Antworten



Definition einer Klasse in LotusScript

Class Company	Klasse
Private strCompanyName As String Private strCompanyNr As String	Membervariablen
Public ContactName As String	
Sub New (strCompanyName As String , strCompanyNr As String) Me .strCompanyName = strCompanyName Me .strCompanyNr = strCompanyNr End Sub ' Contact.New	Konstruktor
Property Get CompanyName As String CompanyName = Me .strCompanyName End Property	Property Get
Property Set CompanyName As String Me .strCompanyName = CompanyName End Property	Property Set
End Class	



Verwenden einer Klasse

'Variable definieren

```
Dim objCompany As Company
```

'Objekt intialisieren

```
Set objCompany = New Company ("Tesla Incorporated", "Tesla")
```

'Setzen der öffentlichen Membervariablen

```
objCompany.ContactName = "Elon Mask"
```

'Setzen der privaten Membervariablen

```
objCompany.CompanyName = "Tesla Germany GmbH"
```

'Zugriff auf die Membervariablen

```
MessageBox objCompany.ContactName & " - " & objCompany.CompanyName & " (" & objCompany.CompanyNr & ") ", 64,  
"Demo"
```



Vererbung

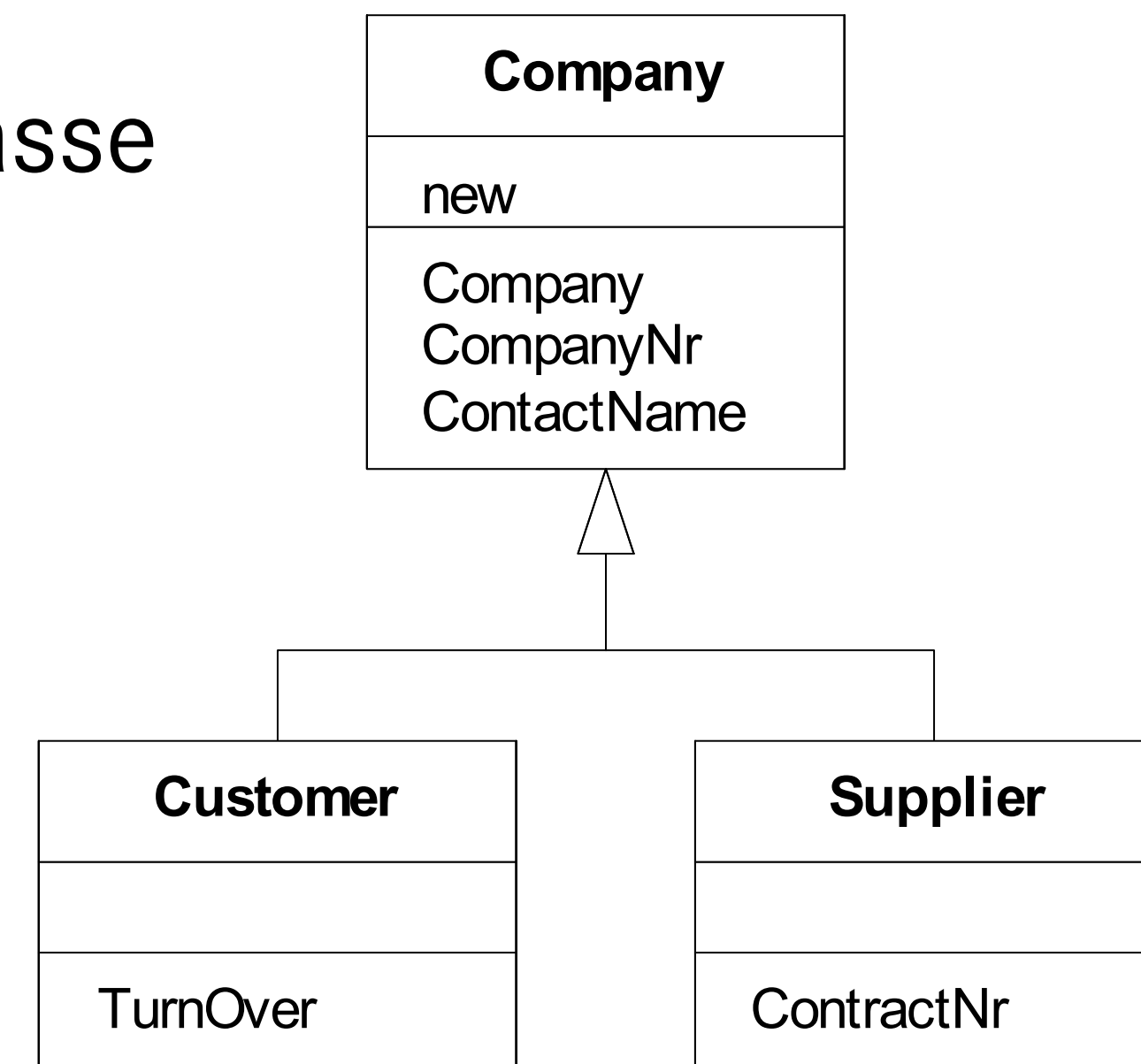
- Übernehmen von Properties und Methoden einer Klasse
- Ergänzung durch eigene Properties und Methoden

```
Class Customer As Company
    Public TurnOver As Double

    Sub New (strCompanyName As String, _
            strCompanyNr As String)

    End Sub 'Customer.New

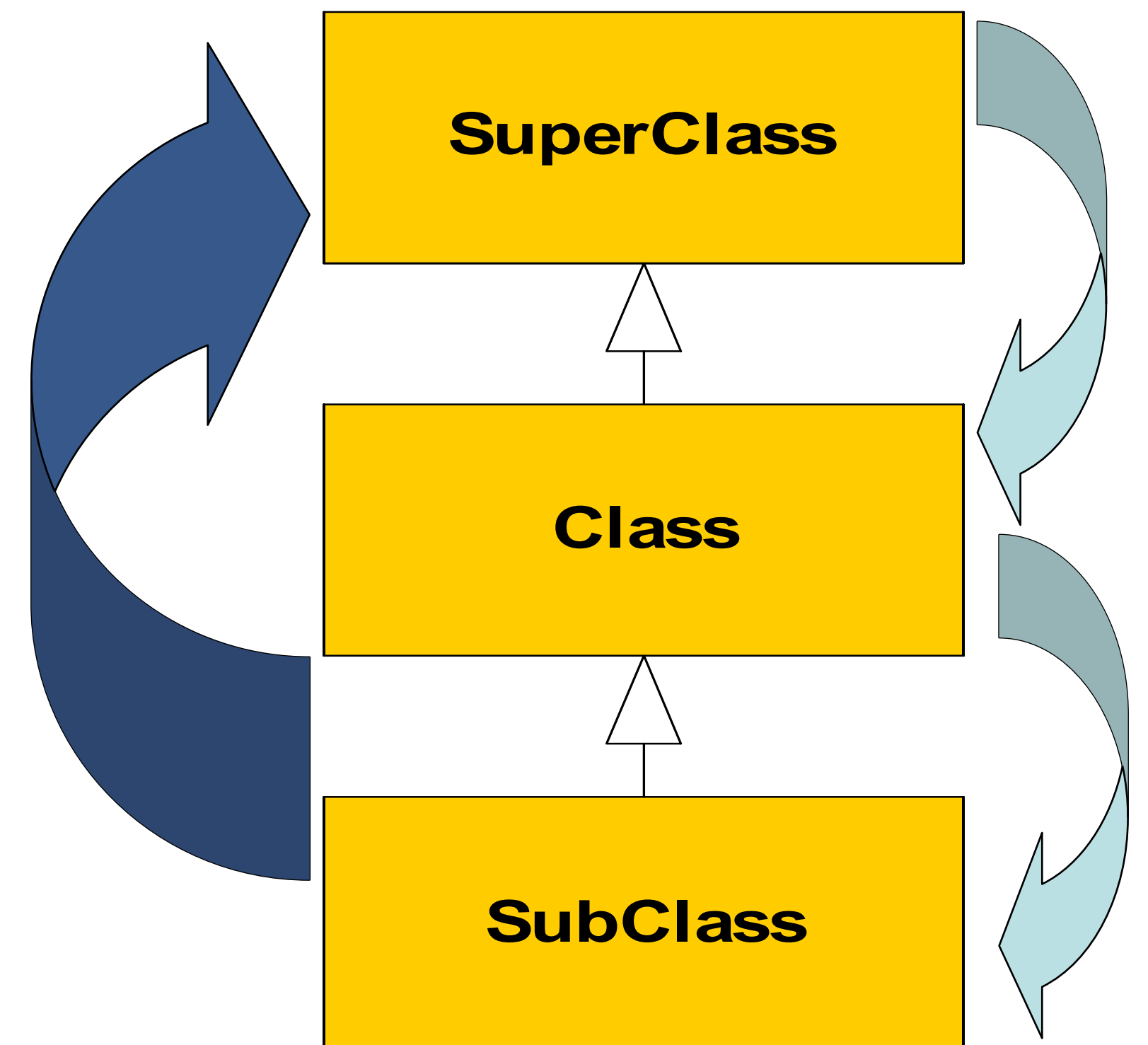
End Class 'Customer
```



Initialisierungsreihenfolge bei Unterklassen

- Es wird zuerst der Konstruktor der Superklasse aufgerufen
- Danach alle weiteren Konstrukteure

```
Set objSubClass = New SubClass
```



Konstruktor überschreiben

- Abgeleitete Klassen können einen von der Elternklasse abweichenden Konstruktor haben.

```
Class Supplier As Company
    Private strContractNr As String

    Sub New (strCompanyName As String, strCompanyNr As String, _
        strContractNr As String),
        Company (strCompanyName, strCompanyNr)

        Me.strContractNr = strContractNr
    End Sub 'Supplier.New
End Class 'Supplier
```

Übergabe der Parameter für die Elternklasse



Methoden und Properties überschreiben

- In der abgeleiteten Klasse können Methoden und Properties mit gleichem Namen definiert werden.
- Die Signature muss gleich bleiben.
- Methoden und Properties der Elternklasse können mit `Elternklasse..MethodenName` aufgerufen werden



Standard Notes-Klassen können nicht abgeleitet werden

- Es ist nicht möglich, die Standard Notes-Klassen abzuleiten!



Objektorientierte Programmierung

- Einführung
- Motivation
- Syntax und Definition
- **Code-Beispiele**
- Fragen und Antworten



Demo - Beispielanwendung

- Demo assono Framework
- Basis Version des Frameworks verfügbar bei OpenNTF.org
 - Licensed under the Apache License, Version 2.0



Code-Beispiel - Höhere Abstraktionsebene

- Statt auf Daten zu arbeiten, wird mit Objekten aus der Anwendungsdomäne gearbeitet

IsDate (doc.GetItemvalue (*"ApprovedOn"*) (0))

vs.

model.isRequestApproved ()



Code Beispiel - Umgang mit Komplexität

- Aufteilung auf kleinere Methoden
 - Erhöht die Lesbarkeit
 - Bessere Wiederverwendbarkeit

```
Public Sub RequestApproval()  
  
    Call modelDoc.ReplaceItemvalue("RequestedOn", now)  
    Call modelDoc.ReplaceItemvalue("RequestedBy", session.Username)  
  
    Call me.SetState(STATE_REQUESTED)  
  
    Call me.SaveWithEvents()  
  
    Call me.InformSupervisorAboutNewRequest()  
  
End Sub ' RequestModel.RequestApproval
```



Code Beispiel - Lesbarkeit des Codes

- Verbergen von Details
- Konzentration auf das Wesentliche

```
Public Sub Initialize(doc As NotesDocument)

    Dim constraint As Constraint

    Call RequestModel..Initialize(doc)

    Set constraint = New NotNullConstraint("Vorgesetzter|Supervisor,
    Reiseziel|TravelDestination, Reiseanlass|TravelReason,
    Reisedatum|TravelDate")
    Call AddConstraint(constraint)

End Sub ' TravelRequestModel.Initialize
```



Code Beispiel - Wiederverwendbarkeit

- Class EMail
 - Eine Klasse zum Versenden von E-Mails
 - entweder regulär oder durch Ablegen in der mail.box
 - Manipulation des Body-Inhalts
- Basis für weitere Ableitungen
 - Class EMailFromTemplate - E-Mails aus einer Vorlage



```

/*****
* This class creates emails either normal or directly in the server's mail.box.
*
* @author Thomas Bahn / Bernd Hort
* @version 2007-08-13
*****/

```

EMail - Class EMail

```

GetMailDB() As NotesDatabase
GetMailDoc() As NotesDocument
GetBodyRTItem() As NotesRichTextItem
GetSender() As String
SetSender(Sender As String) As Variant
GetPrincipal() As String
SetPrincipal(Principal As String) As Variant
GetSendTo() As Variant
SetSendTo(Sendto As Variant) As Variant
GetCopyTo() As Variant
SetCopyTo(Ccto As Variant) As Variant
GetBlindCopyTo() As Variant
SetBlindCopyTo(Bccto As Variant) As Variant
GetSubject() As String
SetSubject(Subject As String) As Variant
GetDirectSend() As Boolean
SetDirectSend(Directsend As Boolean) As Variant
GetAdditionalMailItems() As Variant
AddAdditionalMailItem(Itemname As String, Itemvalue As Variant) As Variant
addAdditionalMailItems(Additionalmailitems As Variant list) As Variant
SetAdditionalMailItems(Additionalmailitems As Variant list) As Variant
AddNewLine(Count As Integer) As Variant
AppendDocLink(Linkto As Variant, Comment As String, Hotspottext As String) As Variant
AppendText(Text As String) As Variant
SetStyle(Stylename As String)
New()
CreateItem(Itemname As String, Itemvalue As Variant, Itemtype As Integer)
Send()
Delete()

```

Code Beispiel - Erweiterbarkeit und Flexibilität

- Vererbung und Überladen von Methoden sind ideal, um
 - Gemeinsamkeiten pflegen
 - Abweichungen zulassen
- Beispiel LeaveRequest.Decline

```
Public Sub Declined(reason As String)

    'Elternmethode aufrufen
    Call RequestModel..Declined(reason)

    'Zusätzlich die Personal-Abteilung informieren
    Call me.InformHRAboutDeclination(reason)

End Sub ' LeaveRequestModelModel.Declined
```



Code Beispiel - Vermeidung von Fehlern

- Durch das „Information Hiding“ sind von den Klassen nur die gewünschten Methoden sichtbar.
- Die interne Struktur kann leicht angepasst werden, ohne das sich das Verhalten nach aussen ändert.
- Vermeidung von Seiteneffekten, weil kein Code ausserhalb der Klasse auf interne Variablen zugreifen kann.



Demo - Beschleunigung der Entwicklung

- Demo
MaskenUtil

Masken Util 

Maske

Server

Datenbank Pfad 

Datenbank Titel

Maske

Felder mit Label bestimmen





Projekte\Konferenzen\DominoCamp-2022\DC22T1S2_oop.nsf
DC22-T1S2 OOP in LotusScript
«Form» Urlaubsantrag (LeaveRequest)
 ja  nein **experimentell!**

Felder

Felder

 [Alle auswählen](#)
 [Auswahl aufheben](#)

☐ Authors|Authors
☐ Readers|Readers
☐ Antragsteller|Requester
☐ Vorgesetzter|Supervisor
☐ Urlaubsziel|TravelDestination
☐ Erster Urlaubstag|FirstVacationDate
☐ Letzter Urlaubsantrag|LastVacationDate
☐ ViewAbstract|ViewAbstract

Maskenliste

Feldliste

Field Validation Code

Class Model Code

Class Controller Code

Controller Setup Code

Uniqueness View Build





Fragen?

Picture Credits

- Unplash

<https://unsplash.com/>

- Pexels

<https://www.pexels.com/>



Stories matter



Photo by [Suzy Hazelwood](#) from [Pexels](#)

Mo. 28.03.2023 - EntwicklerTrack 1 - 11:30 Uhr