

Mobile Business

Realisierung von
Mobile Business Anwendungen II
oder

Wie programmiere ich eine Mobile Business Anwendung?



Dipl.-Ök. Philipp Maske
maske@iwi.uni-hannover.de

Institut für Wirtschaftsinformatik
Leibniz Universität Hannover

Veranstaltung „Mobile Business“, Institut für Wirtschaftsinformatik,
Leibniz Universität Hannover

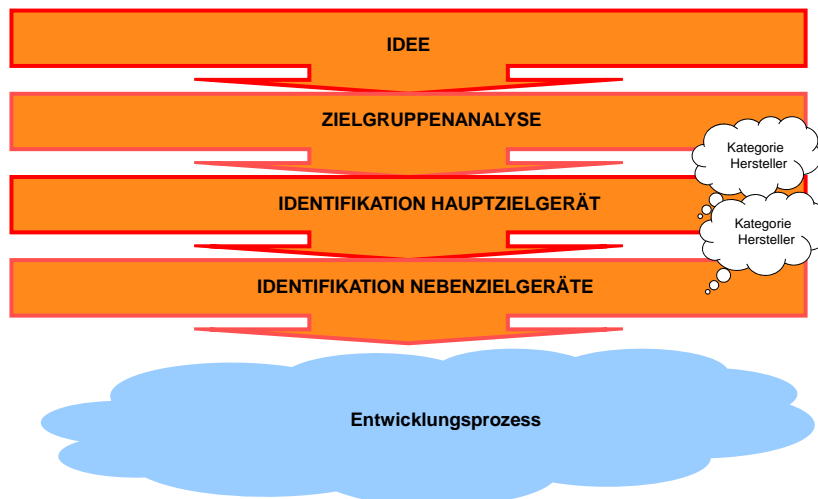
Agenda

- Vorgehensmodelle für mobile Anwendungen.
- CASE-Tools.
- Objektorientierte Programmierung.
- Entwicklungsumgebungen.
- Softwareentwicklung im Team.
- Anwendungsbeispiele (Windows Mobile und Google Android).
- Ausblick.

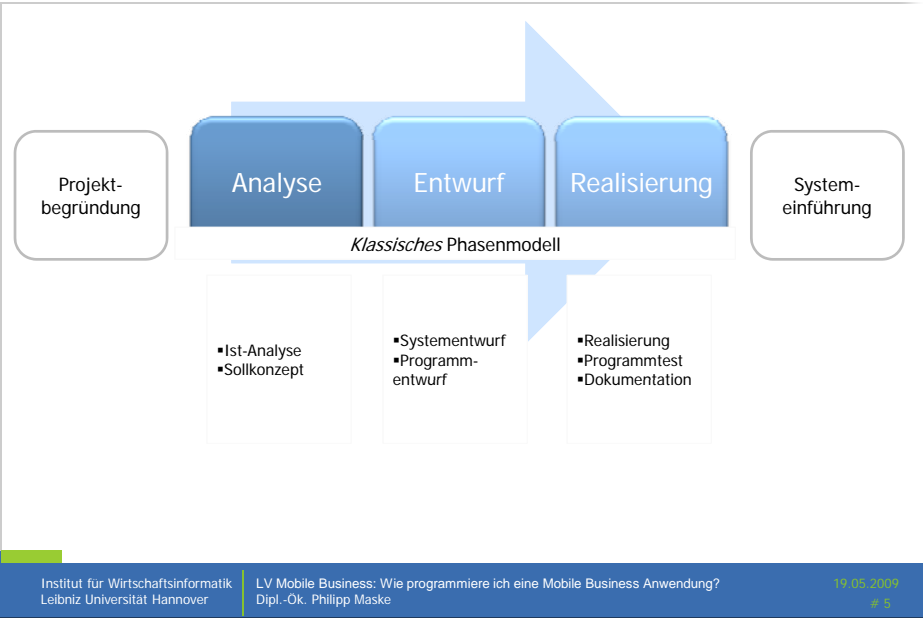
Veranstaltungsplanung

Nr.	Datum	Thema	Dozent
1	31.03.2009	Einführung	Breitner
2	07.04.2009	Vom Electronic Business zum Mobile Business	Maske
3	14.04.2009	Mobile Endgeräte / Funktechnologien	Maske
4	21.04.2009	Location Based Services und Lokalisierung	Maske
5	28.04.2009	Mobile Business Strategien & Personalisierung	Maske
6	05.05.2009	Geschäftsmodelle, Nutzerakzeptanz und Gestaltung von Mobile Business Anwendungen	Maske
7	12.05.2009	Gestaltung und anschließende Realisierung von Mobile Business Anwendungen	Maske
8	19.05.2009	Realisierung von Mobile Business Anwendungen II	Maske
9	26.05.2009	Mobile Payment	Pousttchi
10	09.06.2009	Vom Mobile Business zum Ubiquitous Computing	Wohlers
11	16.06.2009	Mobile Gaming, Mobile Multimedia und Podcasting	Maske
12	23.06.2009	Fallstudie: M-Learning Anwendungsentwicklung	Maske
13	30.06.2009	Klausur	

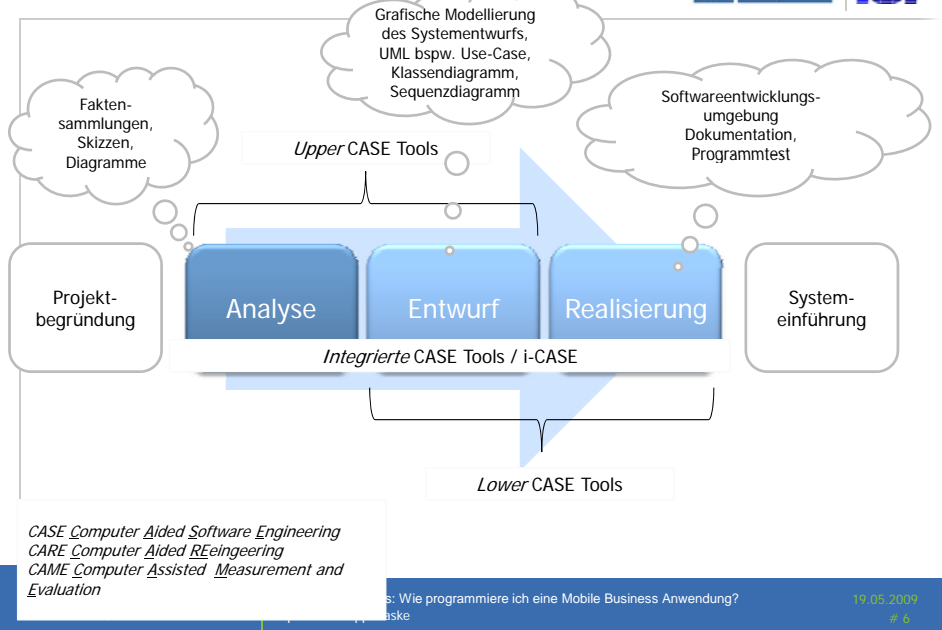
„Kochrezept“



Vorgehensmodelle in der Systementwicklung



CASE Tools



Istanalyse:

- Umfeldanalyse mit dem Ziel,
ein Sollkonzept aufzustellen:
 - Im Sollkonzept wird in einem *Grobkonzept* festgehalten,
 - **was** das Anwendungssystem leisten soll (**Fachentwurf**) und
 - **wie** das Anwendungssystem realisiert werden soll (**IV-Grobentwurf**).

Zur Umfeldanalyse gehören bspw.:

- Nutzerzielgruppen,
- Geräteplattformen,
- Nutzerbedürfnisse,
- Erreichbare Mehrwerte,
- Marktverteilungen und Kosten.

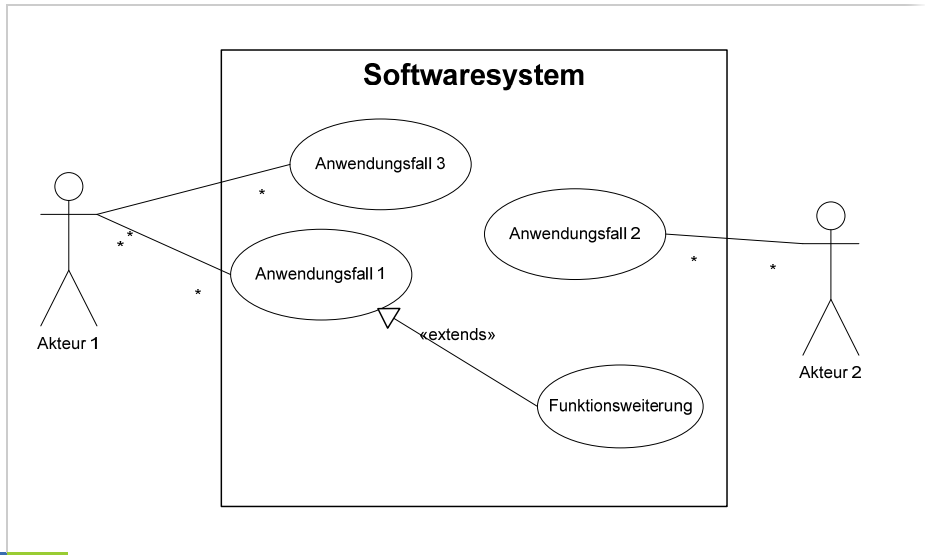
Welche Anwendungsfälle werden unterstützt? → UML Anwendungsfall-Diagramme / (Use Case Diagramme)

**Zeigt das externe Verhalten eines Systems aus Nutzersicht,
Darstellung von:**

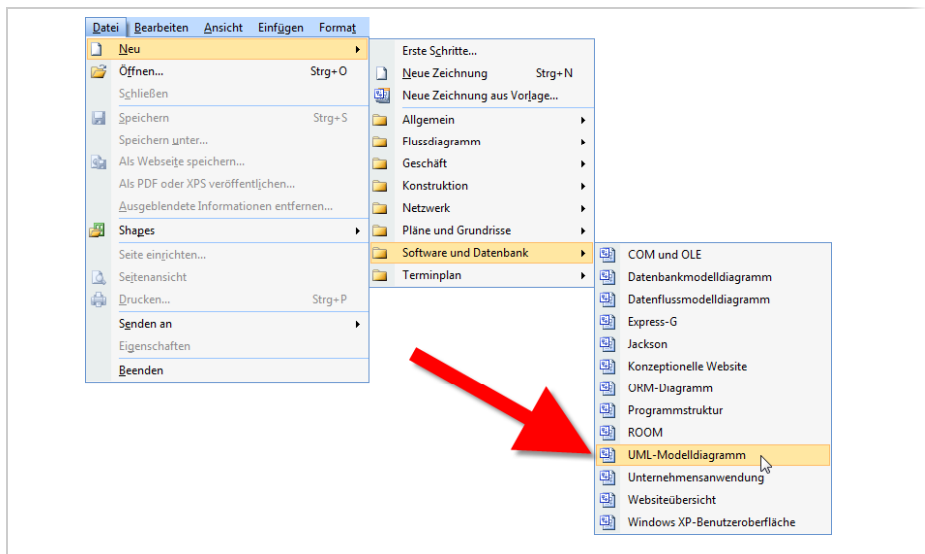
- **Nutzer** (Akteure),
- **Anwendungsfälle** (Use Cases),
- **Beziehungen zueinander.**

- Ein **Nutzer** kann eine **Person** aber auch ein **System** sein.
- Ein **Use Case** kapselt eine in sich geschlossene Sammlung von Aktionen, um der Umwelt eine Dienstleistung (bzw. ein Verhalten) anzubieten.
- **„Was statt wie“:** Ein Use Case zeigt den Auslöser eines Use Case, einzelne Schritte (auch Sonder- und Fehlerfälle) und daran beteiligte externe Personen/Systeme.
→ Es zeigt jedoch nicht welcher Programmcode, welche Klassen bzw. Operationen daran beteiligt sind.

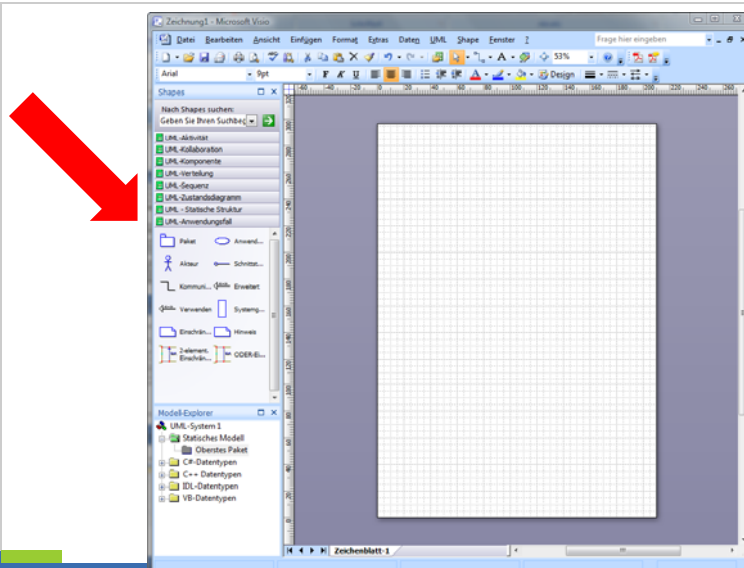
Beispiel für ein UML Use Case Diagramm



Exkurs: Use Case Diagramme mit Visio 2007



Exkurs: Use Case Diagramme mit Visio 2007 (2)



Entwurfsphase

Die Entwurfsphase soll die Voraussetzungen der nachfolgenden Realisierungsphase schaffen.

Die Entwurfsphase beinhaltet drei wichtige Schritte:

- **Schritt 1:** Erstellung eines **Systementwurfs**, wobei eine grundlegende Entscheidung getroffen werden muss, ob der Systementwurf **objektorientiert** oder **strukturiert** erfolgen soll.
- **Schritt 2:** Zusammenstellung einer Programmspezifikation, d. h. detaillierte Vorgaben in Form eines Pflichtenheftes.
- **Schritt 3:** Erarbeitung eines systematischen und möglichst strukturierten Programmentwurfs anhand der Programmspezifikation.

In die Entwurfsphase gehört noch nicht die Umsetzung in Programmcode!

Entwicklung eines Systementwurfs

- Ein Systementwurf kann entweder durch einen Prozess der **Spezialisierung**, bei dem das Gesamtsystem schrittweise in Teilsysteme zerlegt wird (**top-down**) oder durch einen Prozess der Generalisierung erstellt werden. Beim Prozess der Generalisierung (**bottom-up**) werden Teilsystemen zum Gesamtsystem zusammengefügt.
- In einem objektorientierten Systementwurf eignet sich **UML (Unified Modeling Language)** als Diagrammtyp zur Darstellung.
- Für den Systementwurf und die Modellierung von funktionalen Anforderungen ist der **UML-Diagrammtyp Use-Case** vorgesehen.
- Ein Use-Case Diagramm stellt besteht aus einer Menge von Anwendungsfällen und stellt die Beziehungen zwischen Akteuren (**Mensch oder System!**) und Anwendungsfällen grafisch dar.

Programmspezifikation mit einem Pflichtenheft

- Ein Pflichtenheft enthält eine detaillierte verbale Beschreibung, **was** die Anwendung leisten soll (**nicht: wie die Anwendung realisiert werden soll!**).
- Ein Pflichtenheft enthält eine Zielbestimmung, die von der Anwendung erreicht werden sollen. Dabei wird eine Unterteilung zwischen **Musskriterien**, **Wunschkriterien** und **Abgrenzungskriterien** vorgenommen.
- **Musskriterien** müssen von der Anwendung in jedem Fall erfüllt werden.

Die Anwendung muss dabei folgende Anforderungen auf jeden Fall erfüllen (**Musskriterien**):

- ▶ Die Anwendung konfrontiert den Benutzer innerhalb eines Lernprojekts mit Fragen der Typen Multiple-Choice, Single-Choice und Textaufgaben.
- ▶ Die Anwendung wird dem Benutzer bei Bedarf Attachments, insbesondere Grafiken, Sounds und Videos anzeigen.
- ▶ Die Anwendung zeigt dem Benutzer fachlich korrekte Antworten an, falls dieser eine Aufgabe korrekt gelöst hat.
- ▶ Ist ein Lernprojekt vollständig bearbeitet worden, erhält der Benutzer eine Statistik seines Lernerfolgs.

Programmspezifikation mit einem Pflichtenheft (2)

- **Wunschkriterien** sind Anforderungen, die sinnvoll sind, aber nicht unbedingt erfüllt werden müssen.

Folgende Kriterien sollten sinnvollerweise erfüllt werden (**Wunschkriterien**):

- ▶ Die Anwendung lässt sich leicht installieren.
- ▶ Die Anwendung kann unterschiedliche Lernprojekte gleichzeitig verarbeiten.
- ▶ Die Anwendung kann Lernprojekte bei Bedarf online herunterladen.
- ▶ Die Anwendung kann bearbeitete Lernprojekte zwischenspeichern, so dass eine Wiederaufnahme z. B. nach einer Pause möglich ist.
- ▶ Die Anwendung unterstützt Online-Funktionen zum kollaborativen Lernen, z. B. die Teilnahme an einem Diskussionsforum.
- ▶ Die Anwendung zeigt eine grafische Statistik des Lernerfolgs an.
- ▶ Die Anwendung hat eine integrierte Online-Hilfe.

Programmspezifikation mit einem Pflichtenheft (3)

- **Abgrenzungskriterien** sollen von der Anwendung bewusst nicht erfüllt werden.

Folgende Kriterien sollen von der Anwendung bewusst nicht erfüllt werden (**Abgrenzungskriterien**):

- ▶ Die Anwendung bietet keinen Mehrbenutzerbetrieb, d. h. mehrere E-Learner arbeiten asynchron mit derselben Anwendung.
- ▶ Die Anwendung bietet keine Benutzerauthentifikation zum Schutz vor einer unberechtigten Inbetriebnahme durch Dritte bspw. bei Verlust des mobilen Endgeräts. Hierzu sind stattdessen geeignete Schutzvorkehrungen auf Betriebssystemebene vorzunehmen.
- ▶ Die Anwendung unterstützt kein Multithreading, d. h. die scheinbar gleichzeitige Ausführung mehrerer Prozesse. Zum einen stehen Prozessor- und Speicherplatzrestriktionen dieser Anforderung entgegen, zum anderen ist Multithreading für den Betrieb einer mobilen E-Learning Anwendung nicht erforderlich, da sich aus Benutzersicht kein Zeitgewinn ergibt.

Programmspezifikation mit einem Pflichtenheft (4)

- Weiterhin sollte der **Einsatzbereich** der Anwendung genau definiert werden, bspw.
→ „Einsatz im *privaten* Bereich, bewusst *nicht im gewerblichen* Bereich“.
- Auch eine **Zielgruppe** sollte definiert werden, also bspw. „Studenten der Wirtschaftswissenschaften im Grundstudium“.
- Wichtig ist auch, die **Umgebung** zu definieren, auf dem die Anwendung zukünftig laufen soll. Bspw. „Smartphone HTC mit Windows Mobile 2005“ oder „Google Android Smartphone G1“.
- Angabe der benutzten **Entwicklungsumgebung** (z.B. Microsoft Visual Studio .NET 2008, Eclipse).
- Einen Hauptteil des Pflichtenhefts machen die **vorgesehenen Funktionen (Use-Case)** in einer abstrakten Benutzersicht aus.

Programmspezifikation mit einem Pflichtenheft (5)

Beispiel für einen Anwendungsfall (Use-Case):

- Ausgabe einer persönlichen Begrüßung auf dem Handy-Display:
 - Typischer Ablauf:
 1. Eingabe des Namens.
 2. Drücken des Begrüßungsbuttons.
 3. Prüfung, ob Name nicht leer.
 4. Präsentation der persönlichen Begrüßung.
 - Mögliche Alternativen:
 1. Name ist leer.
 2. Ausgabe einer Fehlermeldung.

Programmspezifikation mit einem Pflichtenheft (6)

Die Benutzeroberfläche ist ein entscheidender Faktor für die Akzeptanz durch den zukünftigen Benutzer.

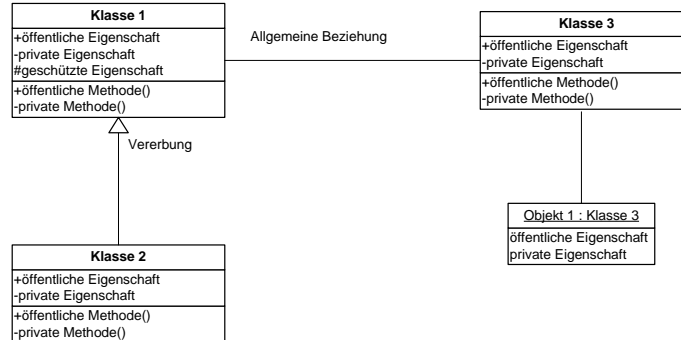
Definition der Anforderungen an die Benutzeroberfläche, bspw.:

- „Die Bildschirmschriftart soll gut lesbar sein“,
- „Durch einen systematischen Einsatz von Farben soll die Orientierung durch den Benutzer erleichtert werden“,
- „Die Gestaltung der Benutzeroberfläche soll auf windowstypische Elemente zurückgreifen, um eine möglichst kurze Eingewöhnungszeit durch den Benutzer zu ermöglichen“.

Erstellung eines Programmentwurfs

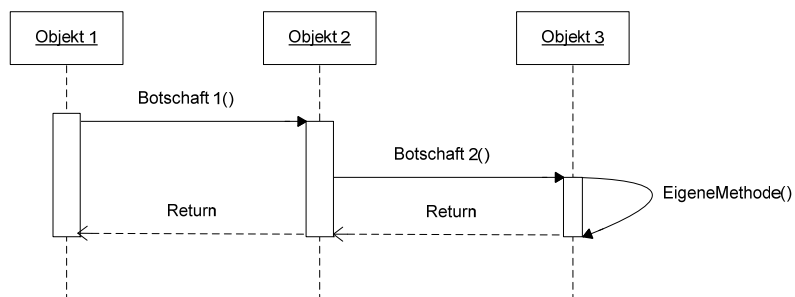
- Anhand des entwickelten Systementwurfs und unter Berücksichtigung des Pflichtenheftes wird ein **detaillierter Programmentwurf erstellt**, der entweder **objektorientiert** oder **strukturiert** erfolgen kann.
- Am Beispiel eines objektorientierten Systementwurfs:
 - Das Gesamtsystem wird in Teilsysteme zerlegt, um in einer gedanklichen Analyse „**Gegenstände**“ zu isolieren. Diese Gegenstände bilden **Klassen**, die anschließend in eine **Klassenhierarchie** geordnet werden (**fachlicher Systementwurf**). Dieser fachliche Systementwurf bildet die Grundlage für den **technischen Systementwurf**, bei dem das softwaretechnische Repertoire der objektorientierten Programmierung zum Einsatz kommt (Vererbung, Verwendung von abstrakten Datentypen, Polymorphie).

Erstellung eines Programmentwurfs



Anhand des entwickelten **Systementwurfs** und unter Berücksichtigung des **Pflichtenheftes** wird ein detaillierter **Programmentwurf** erstellt, der entweder objektorientiert oder strukturiert erfolgen kann.

UML Sequenzdiagramm



Realisierungsphase

- Die Realisierungsphase wird auch **Implementierungsphase** genannt.
- Die Realisierungsphase beinhaltet die Vorgänge:
 - **Programmierung** der Anwendung, und
 - einen anschließenden **Programm- und Systemtest**.
- Damit Programme besser lesbar sind, sollte eine ausführliche Selbstdokumentation (Inlinedokumentation im Programmcode) erstellt werden. Diese besteht in der Regel aus speziellen Kommentarzeilen innerhalb des Programmcodes, die beim Kompilieren ignoriert werden, aber die Lesbarkeit des Quelltextes erhöhen.
- In einem anschließenden Programmtest wird überprüft, ob die Anwendung die gestellten Aufgaben zufrieden stellend löst und das Programm stabil läuft.
- Zum Programmtest sollte mindestens eine, möglichst mehrere, Testplattformen definiert werden.
- Es gibt Software-Tools, die beim Programmtest unterstützen können.

Programmdokumentation in Eclipse: Code-Kommentare

```
40 this.r = r;
41 }
42
43 /**
44  * Enthält den TextView des aktuellen Formulare
45  */
46 public TextView tv;
47
48 /**
49  * Enthält den Button des aktuellen Formulare
50  */
51 public Button bt;
52
53 /**
54  * Enthält das Texteingabefeld
55  */
56 public EditText et;
57
58
59 /**
60  * Wird bei Erstellen des Formulare aufgerufen
61  * @param icicle Enthält eine Ansammlung von Werten des aktuellen Formulare
62  */
63 @Override
64 public void onCreate(Bundle icicle) {
65     super.onCreate(icicle);
66     LinearLayout ll = new LinearLayout(this);
67     LinearLayout.LayoutParams llp = new LinearLayout.LayoutParams(LinearLayout.LayoutParams.MATCH_PARENT,
68     LinearLayout.LayoutParams.WRAP_CONTENT);
69     ll.setLayoutParams(llp);
70     ll.setOrientation(LinearLayout.VERTICAL);
71
72     tv = new TextView(this);
73     tv.setText("Name");
74     tv.setId(R.id.guessText);
75
76     bt = new Button(this);
77     bt.setText("GUESS WIECH");
```

Programmdokumentation in Eclipse: Dokumentation mit Javadoc

The screenshot shows the Eclipse IDE's Javadoc viewer for the class `TestAndroid`. The left pane shows the project structure with `TestAndroid` selected. The main pane displays the following information:

- Package:** `de.hannover.universitaet.iwi.androidtest`
- Class:** `TestAndroid`
- Java Language Objects:**
 - `Activity`
 - `L.de.hannover.universitaet.iwi.androidtest.TestAndroid`
- Code Snippet:**

```
public class TestAndroid
    extends Activity
```
- Description:** Ein Testprogramm für Google Android in Java. Geschrieben für die Veranstaltung Systementwicklung & Softwareengineering.
- Version:** 0.2
- Author:** Philipp Maske
- Field Summary:**
 - `button` `int`: Enthält den Button des aktuellen Formulars
 - `editText` `EditText`: Enthält das Texteingabefeld
 - `textView` `TextView`: Enthält den TextView des aktuellen Formulars
- Constructor Summary:** (Section header visible)

Gängige Entwicklungsumgebungen und Programmiersprachen

Name	Hersteller Lizenz	Verwandtschaft Einflüsse	Editoren
Java ME Java	Sun Kommerziell Open-Source	C#, C++, Smalltalk	Eclipse Netbeans (Sun) Jbuilder (Borland)
C#	Microsoft Kommerziell Open-Source	Java, C++, Delphi	Visual Studio (MS) SharpDevelop MonoDevelop
Visual Basic	Microsoft Nur kommerziell	Basic (evtl. Java, C#)	Visual Studio (MS) SharpDevelop
PHP	Open-Source	C	Zend Studio PDT (PHP Developer Tools mit Eclipse)
Android (Java)	Open Handset Alliance Open-Source	Java	Eclipse (mit Plugin)
iPhone Cocoa	Apple Kommerziell	Smalltalk / C	Xcode IDE (Apple)

HTML (Hypertext Markup Language)

- Definiert von Tim-Barners Lee in 1989 (CERN), weiterentwickelt vom World Wide Web Konsortium (W3C).
- Textbasierte Auszeichnungssprache (nicht: Seitenbeschreibungssprache; nicht: Programmiersprache!) zur *Strukturierung* von Dokumenten.
- Charakteristisch: Hyperlinks!
- Texte werden durch Tags mit Markups versehen und strukturiert.
- Tags haben (meistens) einen Eröffnungstag und ein Schlusstag.
- Es geht um beschreibende Auszeichnung, keine prozedurale oder darstellende Auszeichnung.
- Ziel ist nicht, eine Darstellung zu beschreiben, die den exakten Vorstellungen des Autors entspricht, sondern eher den technischen Gegebenheiten und Bedürfnissen des Lesers.
- Vergleichbares Konzept zu TeX/LaTeX.
- Wer sich den Quelltext von Webseiten anschauen möchte, wählt im Browser bspw. „Quelltext anzeigen“.

Beispiel: Quelltext der Seite „Mobile Business“ (Auszug)

```
<table height="298" width="1046" border="0" style="width: 1046px; height: 298px;">
  <tbody>
    <tr>
      <td><strong>Nr.</strong></td>
      <td><strong>Datum</strong></td>
      <td><strong>Thema</strong></td>
      <td><strong>Folien (color)<br />
      </strong></td>
      <td><strong>Folien (sw)<br />
      </strong></td>
      <td><strong>Lecturnity Video<br />
      </strong></td>
      <td><strong>MP4-Video</strong></td>
      <td><strong>MP3-Audio </strong></td>
    </tr>
    <tr>
      <td>1.</td>
      <td>31.03.2009</td>
      <td>Einführung</td>
      <td><a href="images/stories/PDF/m-business310309_co.pdf" target="_blank" title="Nr. 1: Einführung in den Mobile Business vom 31.03.2009 (Prof. Dr. Michael H. Breitner) - farbige Version">PDF (1,2 MB)</a></td>
      <td><a href="images/stories/PDF/m-business310309_sw.pdf" target="_blank" title="Nr. 1: Einführung in den Mobile Business vom 31.03.2009 (Prof. Dr. Michael H. Breitner) - schwarz-weiss-Version"><span style="color: rgb(0, 102, 204);">PDF (2 MB)</span></a></td>
      <td><a href="http://stream.mml.uni-hannover.de/Video-Files/iwi/WirtInf_SS09/M-Business/09-03-31/09-03-31.html" target="_blank" title="Lecturnity-Video zu Veranstaltung Nr. 1: Einführung in den Mobile Business vom 31.03.2009 (Prof. Dr. Michael H. Breitner)"><span style="color: rgb(0, 102, 204);">LECTURNITY-Format</span></a></td>
      <td><a href="http://stream.mml.uni-hannover.de/Video-Files/iwi/WirtInf_SS09/M-Business/09-03-31/09-03-31.mp4" target="_blank" title="MP4-Video zu Veranstaltung Nr. 1: Einführung in den Mobile Business vom 31.03.2009 (Prof. Dr. Michael H. Breitner)">MP4-Format</a></td>
      <td><a href="http://stream.mml.uni-hannover.de/Video-Files/iwi/WirtInf_SS09/M-Business/09-03-31/09-03-31.mp3" title="MP3-Audio zu Veranstaltung Nr. 1: Einführung in den Mobile Business vom 31.03.2009 (Prof. Dr. Michael H. Breitner)">MP3-Format</a></td>
    </tr>
  </tbody>
</table>
```

CSS – Cascading Style Sheets

- Legt fest, wie durch Auszeichnungssprachen beschriebene Inhalte dargestellt werden (v.a. HTML / XML).
- CSS können für verschiedene Ausgabemedien festgelegt werden (Bildschirm, Drucker, Projektion, Sprache, Mobilgerät,...).
- „Überlappende“-Stylesheets deshalb, da Style-Eigenschaften vererbt/modifiziert werden können.
- Es lassen sich Angaben definieren bspw. zu Farben, Schriften, Abständen, Rahmen, feste Positionierungen, Absätzen etc.
- Beim Erstellen von HTML muss darauf geachtet werden, dass Elemente, die gleich aussehen sollen, auch gleich ausgezeichnet werden.
- In Praxi: Browser haben immer noch Probleme, manche CSS-Eigenschaften korrekt umzusetzen.

Name	Gerätebeschreibung
all	Gilt für alle Ausgabegeräte.
speech	Sprachbasierte Ausgabegeräte (bspw. Screenreader).
braille	Blindenschriftfähige Ausgabegeräte.
embossed	Blindenschriftfähige Drucker.
handheld	Handheld-Geräte (bspw. Palmtops, Windows Mobile, Blackberry, ...) – nicht iPhone!
print	Drucker
projection	Video-Beamer, Projektoren.
screen	Bildschirm.
tty	Darstellungsgeräte, die feststehende Zeichentypen haben (bspw. Terminals, Fernschreibe, Mobiltelefone) ~ “Teletypewriter“
tv	TV Geräte.

Beispiel: CSS-Quelltext der Seite „Mobile Business“ (Auszug)

```
/* VERSCHIEDENE EINSTELLUNGEN */
/* Datum, Autor*/
.contentheading {
  /*color : #000000;
  font-size: 1.3em;
  font-weight: bold;
  text-decoration:none*/
  margin-top: 0px;
  font-weight: bold;
  font-size: 1.3em;
  background-color: #efefef;
  color: #3f3f3f;
  min-height: 2.2em;
  height: 2.2em;
  width: 40em;
  padding-top: 2px;
  padding-right: 0px;
  padding-bottom: 2px;
  padding-left: 11px;
  border-left-width: 6px;
  border-left-style: solid;
  border-left-color: #3a6daf;
  border-left-width-ltr-source: physical;
  border-left-width-rtl-source: physical;
  border-left-style-ltr-source: physical;
  border-left-style-rtl-source: physical;
  border-left-color-ltr-source: physical;
  border-left-color-rtl-source: physical;
  border-top-width: 1px;
  border-top-style: solid;
  border-top-color: #d1d1d1;
}

.createdate {
  color: #99;
  font-size: 0.8em;
  text-align: left;
  line-height: 0.8em;
}

/* Format für "Written by:..." */
.smalldark {
  color: #111111;
  font-size: 8pt;
}

/* Format für Umfrageergebnisseite, für " Number of Voters" */
/* Jans Tabellenformatierungen */

.mitarbeiter {
  border: 1px solid #04068c;
  min-width: 700px;
}

.mitarbeiter td {
  border: 1px solid #04068c;
  padding: 5px;
}

.forschung {
  border: 0px;
  font-size: 0.8em;
  vertical-align: middle;
}

.tabelle {
  border: 1px solid #04068c;
  border-collapse: collapse;
  font-size: 90%;
  line-height: 140%;
  width: auto;
}

.tabelle td {
  border: 1px dotted #04068c;
  padding: 3px;
  text-align: left;
}
```

Weitere relevante Sprachen (3)

JavaScript

- Skriptsprache, wird fast immer zur DOM-Manipulation von HTML-Dokumenten in Browsern genutzt.
- Hieß ursprünglich Mocha, dann LiveScript und schließlich aus Marketinggründen JavaScript.
- Ausführung der Skriptsprache auf dem Client (bspw. Browser).
- Dynamisch typisierte, objektorientierte aber klassenlose Sprache.
- In der Syntax Ähnlichkeiten zu C (ggf. Java) – trotz Namensähnlichkeit kaum Gemeinsamkeiten mit Java.
- Sicherheit wird gewährleistet durch „SandBox“-Prinzip.
- Für Webanwendungen stehen mittlerweile vielfältige Bibliotheken bereit.
- JavaScript bildet die Grundlage für AJAX (Asynchronous JavaScript and XML).

Beispiel: JavaScript-Quelltext der Seite „Mobile Business“ (Auszug)



```
function getTextNode(node) {
    var txt = "";
    if (node.nodeType == 3) /* Text Node test */
        txt += node.nodeValue;
    for(var i=0; i < node.childNodes.length; i++) {
        txt += getTextNode(node.childNodes[i]);
    }
    return txt;
}
```

Exkurs: Grundlagen der objektorientierten Programmierung



- Die objektorientierte Programmierung basiert auf den Grundprinzipien:
- Datenkapselung und Objektbildung,
- Klassenbildung und Vererbung und
- Botschaftenkommunikation und Polymorphismus.

Datenkapselung und Objektbildung:

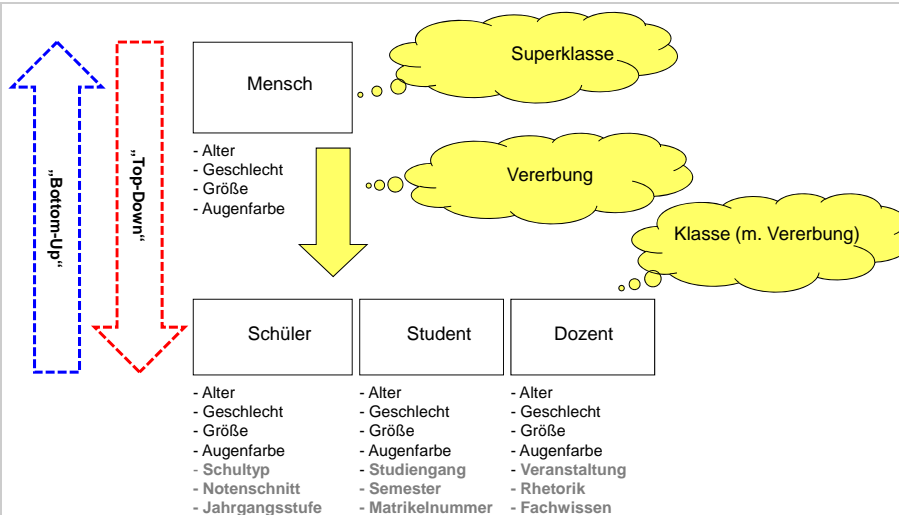
- Unter **Objekten** werden **Personen**, **Gegenstände** und **abstrakte Begriffe** zusammengefasst.
- Objekte besitzen **Eigenschaften** und **Verhalten** (ausgedrückt durch Methoden).
- Datenkapselung bedeutet, dass Eigenschaften eines Objekts nur die die Methoden des Objekts, nicht aber von Außen verändert werden können.

Klassenbildung und Vererbung:

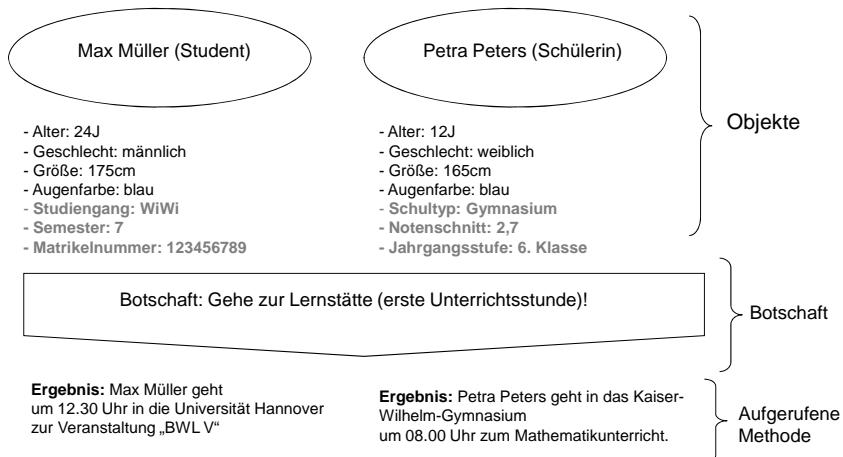
- **Klassen** werden durch das **Zusammenfassen von Objekten** mit den gleichen Eigenschaftstypen gebildet. Die Eigenschaftswerte müssen aber nicht zwangsläufig gleich bleiben. Bestimmte Eigenschaften können von einer übergeordneten Superklasse vererbt werden.

Botschaftenkommunikation und Polymorphismus:

- Zwischen den Objekten gibt es **dynamische Beziehungen**, indem die Objekte **untereinander Botschaften austauschen**. Dabei fungiert jeweils ein Objekt als Sender und ein anderes Objekt als Empfänger.
- Jede Botschaft enthält den Namen des Empfängers, einen Selektor, der die auszuführende Operation angibt, und evtl. optionale Parameter.
- **Polymorphismus bedeutet**, dass die gleiche Botschaft **bei unterschiedlichen Objekten einer Klasse unterschiedliche Operationen angibt**.



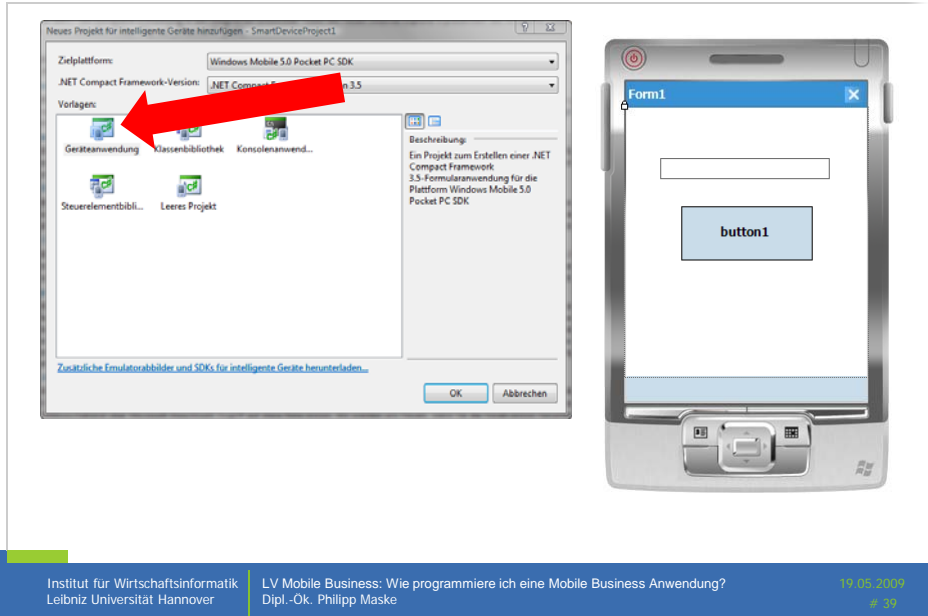
Objektorientierung Programmierung: Objektinstanzen und Polymorphie



Beispiel: Erstellung von Klassen in Visual Studio 2008 – C#

The screenshot shows the Visual Studio 2008 interface. The 'Datei' menu is open, and the 'Projekt...' option is highlighted with a red arrow. The 'New Project' dialog box is open, showing the 'Intelligentes Gerät' template selected. A red arrow points to the 'Intelligentes Gerät' template. The dialog box also shows the 'Projektname' field set to 'MobileBusiness2009test' and the 'Speicherort' field set to 'C:\Users\PhilippMaske\Documents\Visual Studio 2008\Projects\Project1'. The 'Projektmappe' is set to 'Neue Projektmappe erstellen' and the 'Projektmappeverzeichnis erstellen' checkbox is checked.

Beispiel: Erstellung von Klassen in Visual Studio 2008 – C# (2)



Neues Projekt für intelligente Geräte hinzufügen - SmartDeviceProject1

Zielplattform: Windows Mobile 5.0 Pocket PC SDK

.NET Compact Framework-Version: .NET Compact Framework 3.5

Vorlagen:

- Geräteanwendung
- Klassenbibliothek
- Konsolenanwend...
- Steuerelementbibli...
- Leeres Projekt

Beschreibung:
Ein Projekt zum Erstellen einer .NET Compact Framework 3.5-Formulanwendung für die Plattform Windows Mobile 5.0 Pocket PC SDK

OK Abbrechen

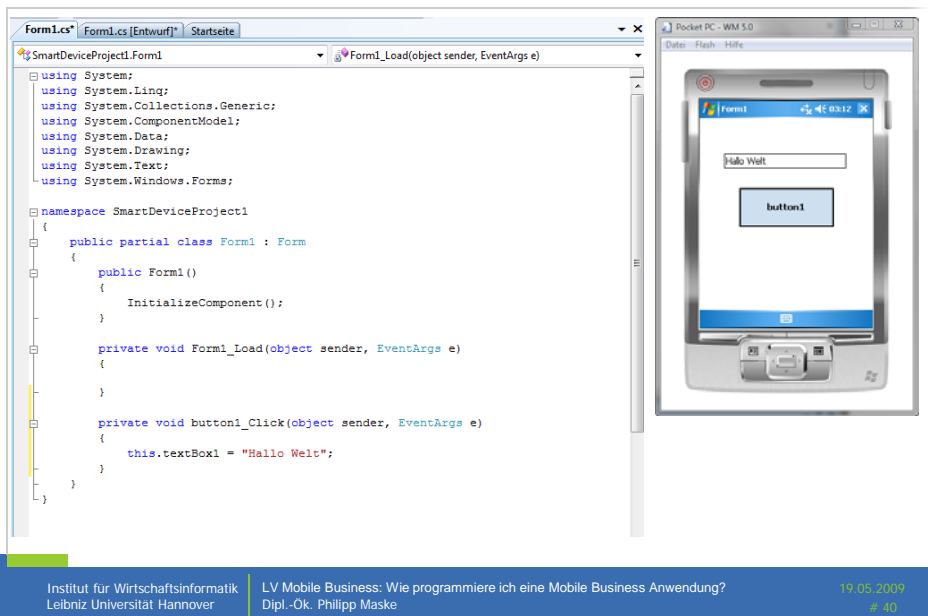
Zusätzliche Emulatorabbildner und SDKs für intelligente Geräte herunterladen...

Institut für Wirtschaftsinformatik
Leibniz Universität Hannover

LV Mobile Business: Wie programmiere ich eine Mobile Business Anwendung?
Dipl.-Ok. Philipp Maske

19.05.2009
39

Beispiel: Erstellung von Klassen in Visual Studio 2008 – C# (3)



```
using System;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace SmartDeviceProject1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.textBox1 = "Hallo Welt";
        }
    }
}
```

Pocket PC - WM 5.0

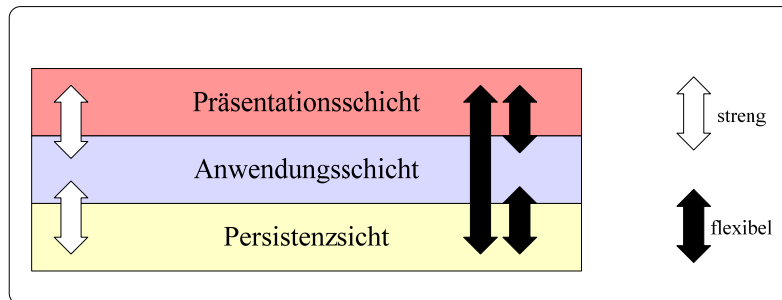
Datei Flash Hilfe

Institut für Wirtschaftsinformatik
Leibniz Universität Hannover

LV Mobile Business: Wie programmiere ich eine Mobile Business Anwendung?
Dipl.-Ok. Philipp Maske

19.05.2009
40

Kommunikation in der 3-Schichten Architektur

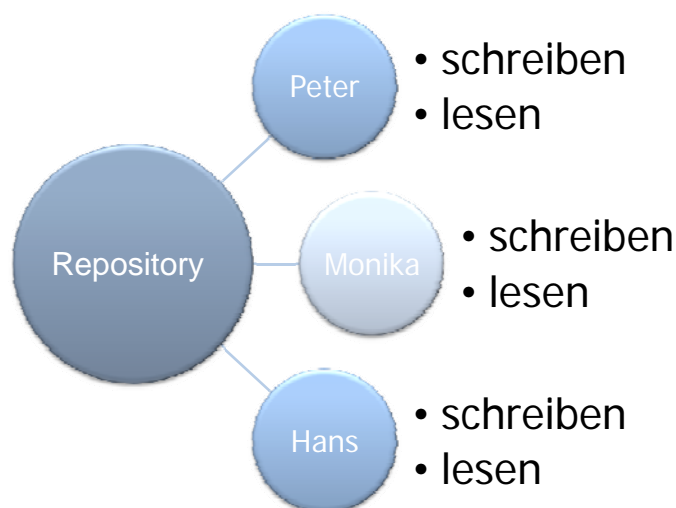


- In der Regel erfordert die Systemeinführung einen erfolgreichen Durchlauf der Testphase, einen förmlichen Systemtest und einen erfolgreich abgeschlossenen Abnahmetest.
- Die Verantwortung liegt z. B. beim Leiter des IT-Projekts.

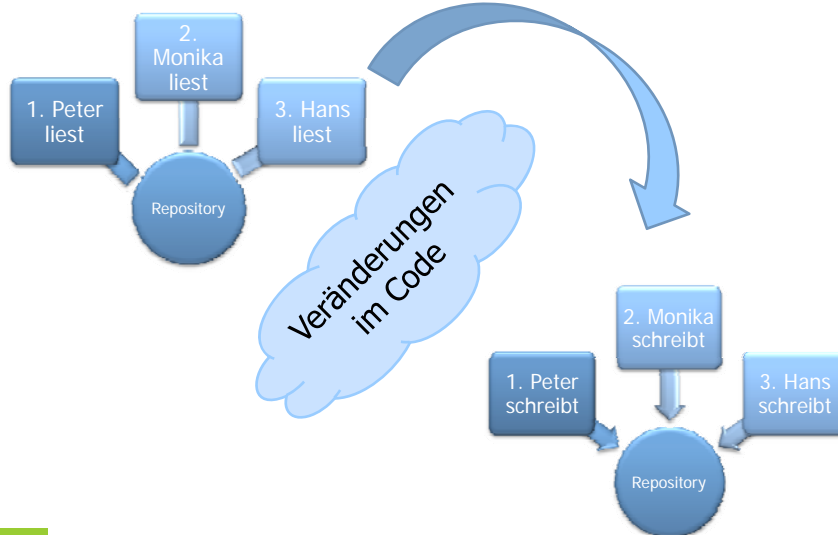
- Die Bearbeitung der Phasen der Softwareentwicklung kann durch Programme unterstützt werden, man spricht von **CASE-Tools** (Computer Aided Software Engineering Tools).
- CASE-Tools für die **frühen Phasen** des Softwareengineerings (von der Analyse bis zum Entwurf) werden auch Upper-CASE-Tools genannt.
- CASE-Tools für die **späten Phasen** des Softwareengineerings (vom Entwurf bis zum Test) werden auch Lower-CASE-Tools genannt.
- Deckt ein CASE-Tool **alle Phasen** des Softwareengineerings ab, so wird auch von einem I-CASE-Tool gesprochen.

Weitere Tools

- CARE-Tool (Computer Aided Reengineering Tool): Wird für die Softwarewartung benötigt, falls aus gegebenem Programmcode der zugehörige Entwurf oder die zugehörige Dokumentation abgeleitet werden soll.
- CAME-Tool (Computer Assisted Measurement and Evaluation Tool): Hilft bei der ingenieurmäßigen Softwareentwicklung als Analyse-, Meß- oder Bewertungstool.



Das Problem gemeinsam genutzter Dateien



Die „Modify-Lock-Unlock“ Lösung

Arbeitsschritt 1:

- (1) Peter liest eine aktuelle Kopie aus dem Repository.
- (2) Gleichzeitig sperrt er den Lese- und Schreibzugriff auf das Repository für andere (**Lock**).
- (3) Peter nimmt Veränderungen am Code vor.
- (4) Peter überträgt die Änderungen ins Repository (**Modify**) und entfernt die Zugriffssperre (**Unlock**).

Arbeitsschritt 2:

- (1) Monika liest die von Peter aktualisierte Version als Kopie aus dem Repository.
- (2) Monika sperrt den Lese- und Schreibzugriff (Lock).
- (3) Monika nimmt Veränderungen am Code vor.
- (4) Monika überträgt die Änderungen ins Repository (**Modify**) und entfernt die Zugriffssperre (**Unlock**).

Arbeitsschritt 3:

- (1) Vice versa für Hans.

Probleme der „Lock-Modify-Unlock“ Lösung

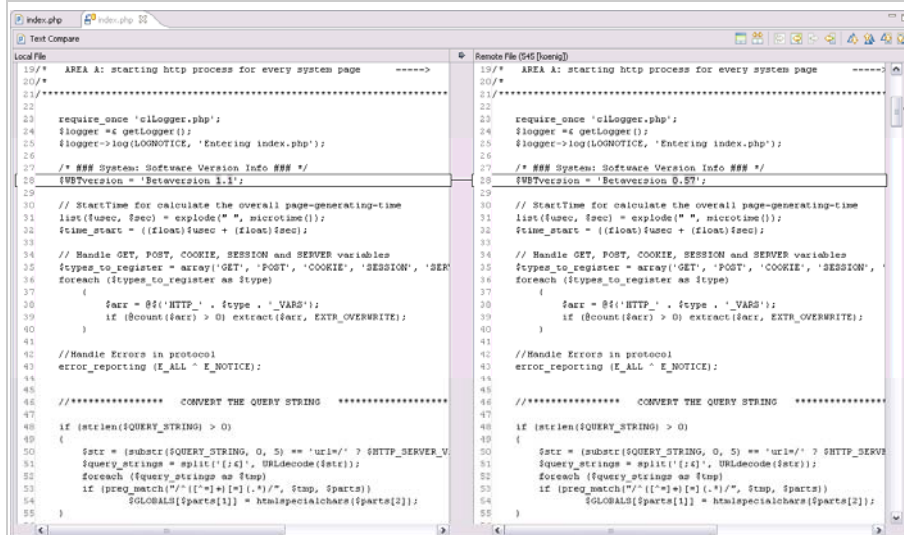
- **Locking kann administrative Probleme verursachen:** Bspw. setzt Peter einen Lock und vergisst diesen...die anderen Teamkollegen warten vergeblich...und dann fährt Peter noch in einen ausgiebigen Urlaub...
- **Locking verursacht unnötige Serialisierung der Arbeitsabläufe:** Bspw. möchte Peter nur den Anfang einer Datei und Monika das Ende einer Datei bearbeiten. Beide könnten gefahrlos nebeneinander arbeiten. Ein Lock verhindert gemeinschaftliches Arbeiten an gleicher Datei.
- **Locking vermittelt einen falschen Eindruck von Sicherheit:** Bspw. bearbeitet Peter Datei A und Monika Datei B. Beide wiegen sich in Sicherheit – jedoch hängt Datei B von Datei A ab.

Die „Copy-Modify-Merge“ Lösung

- (1) Peter und Monika lesen jeweils eine Kopie aus dem Repository.
- (2) Beide bearbeiten ihre individuelle Kopie.
- (3) Monika spielt die bearbeitete Version ins Repository zurück.
- (4) Nun spielt Peter seine Version ins Repository zurück und erhält eine Warnung, dass sein Kopie nicht mehr aktuell ist (Conflict).
- (5) Peter lässt sich die Änderungen von Monika anzeigen und wählt manuell die passende aus. Hieraus ergibt sich ein neues Dokument, dass beide Änderungen kombiniert (Merge).
- (6) Peter spielt die neue Version ins Repository zurück.
- (7) Ab diesem Zeitpunkt können weitere Änderungen vorgenommen werden.

→ **Wichtig:** Das Merging lässt sich nicht automatisch durchführen, es ist immer ein manueller Eingriff nötig!

Merging mit Eclipse



```
19/* AREA A: starting http process for every system page ----->
20/*
21/*-----
22
23 require_once 'cilogger.php';
24 $logger => cilogger();
25 $logger->log(LOGNOTICE, 'Entering index.php');
26
27 /* ### System: Software Version Info ### */
28 $WBVersion = 'Betaeversion 1.1';
29
30 // StartTime for calculate the overall page-generating-time
31 list($usec, $sec) = explode(" ", microtime());
32 $time_start = ((float)$usec + (float)$sec);
33
34 // Handle GET, POST, COOKIE, SESSION and SERVER variables
35 $types_to_register = array('GET', 'POST', 'COOKIE', 'SESSION', 'SER
36 foreach ($types_to_register as $type)
37 {
38     $sarr = @($_HTTP_ . $type . ' VARS');
39     if (@count($sarr) > 0) extract($sarr, EXTR_OVERWRITE);
40 }
41
42 //Handle Errors in protocol
43 error_reporting (E_ALL ^ E_NOTICE);
44
45
46 //***** CONVERT THE QUERY STRING *****
47
48 if (strlen($QUERY_STRING) > 0)
49 {
50     $sarr = (substr($QUERY_STRING, 0, 5) == 'url=' ? $HTTP_SERVER_V
51     $query_strings = split('[:&]', URLdecode($str));
52     foreach ($query_strings as $tmp)
53         if (preg_match("/^([*]*)=([*]*)$/", $tmp, $parts))
54             $GLOBALS[$parts[1]] = htmlspecialchars($parts[2]);
55 }
56
```

Exkurs: Fortgeschrittene Programmierung mit PHP

- PDT Developer Tools (kostenlos)
<http://www.zend.com/de/community/pdt>
- Zend Studio for Eclipse (kostenpflichtig)
<http://www.zend.com/de/products/studio/eclipse/>
- Hervorhebenswerte Eigenschaften:
 - Codeervollständigung,
 - Debugging,
 - Stacktracing,
 - Performancetests,
 - Refactoring,
 - Codeanalyse (PHP, HTML und JavaScript).

Realisierung eines Prototypen für Google Android mit Eclipse



Eclipse ist ein OpenSource Framework zur Entwicklung von Software vieler verschiedenen Arten. Eclipse selbst basiert auf Java Technologie.

Eclipse ist der Nachfolger von IBM Visual Age for Java 4.0, seit 2001 quelloffen und seit 2004 weiterentwickelt durch die rechtlich eigenständige Eclipse Foundation.

Eclipse ist bekannt geworden als IDE für Java, unterstützt aber vielfältige Programmiersprachen (bspw. Java, C++, C#, PHP, JavaScript,...).

Durch eine offene, plug-in basierte Struktur können neue Entwicklungsaufgaben hinzugefügt werden.

Eclipse eignet sich daher besonders als:

- **Lower-CASE Tool** (UML Modellierung, Implementierung, Dokumentation, Programmtest),
- **CAME Tool** (Profiling, Software-Metriken, Junit-Testing),
- **CARE Tool** (Code to model)

Realisierung eines Prototypen für Google Android mit Eclipse (2)



Download von Eclipse bspw. 3.3 von:

<http://www.eclipse.org/downloads/>

Download des Google-Android SDK von:

<http://code.google.com/android/download.html>

- Das Google Android Software Development Kit enthält alle notwendigen Java Klassen, eine online Dokumentation und einen Emulator zum Testen der Anwendungen auf dem PC.

Download des JDK 6 (evtl. auch JDK 5) von:

<http://java.sun.com/javase/downloads/index.jsp>

Installation des Android Plugins for Eclipse, gemäß Anleitung von:

<http://code.google.com/android/intro/installing.html#installingplugins>

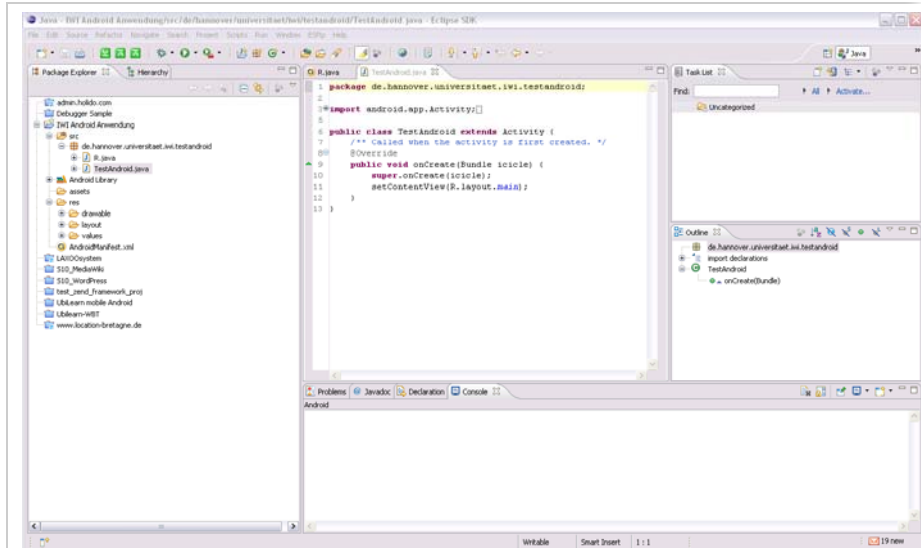
Weiterführende Hilfe (englisch) verfügbar unter:

<http://code.google.com/android/intro/index.html>

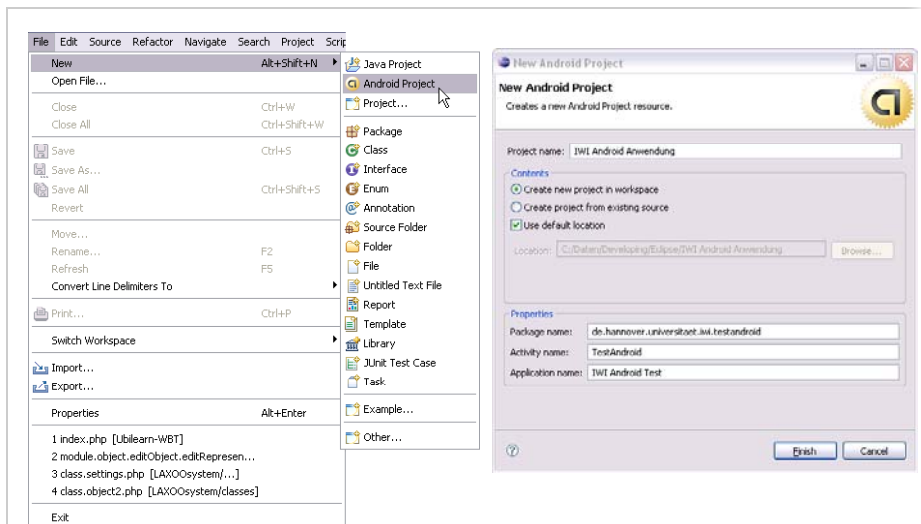
Omondo Eclipse UML (optional):

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

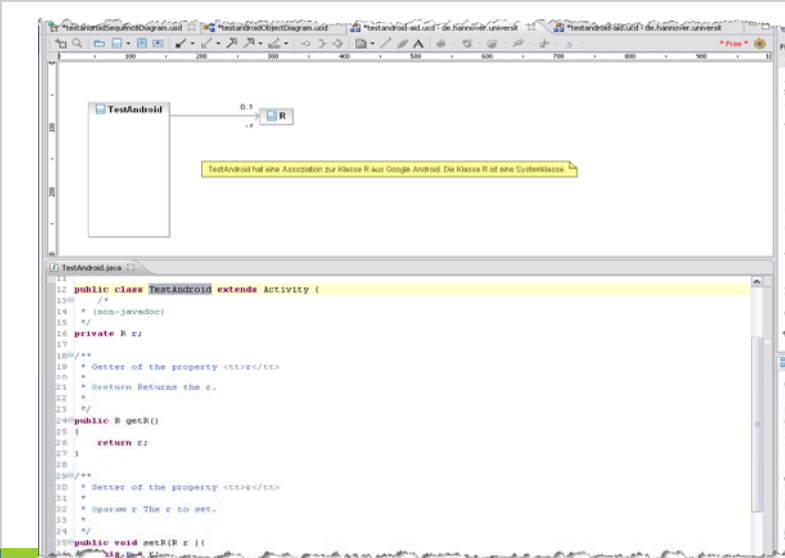
Realisierung eines Prototypen für Google Android mit Eclipse (3)



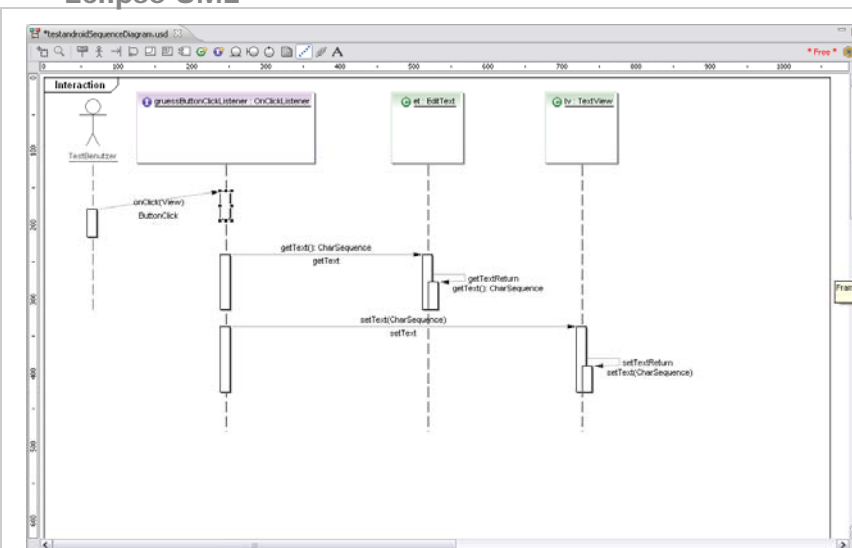
Realisierung eines Prototypen für Google Android mit Eclipse (3)



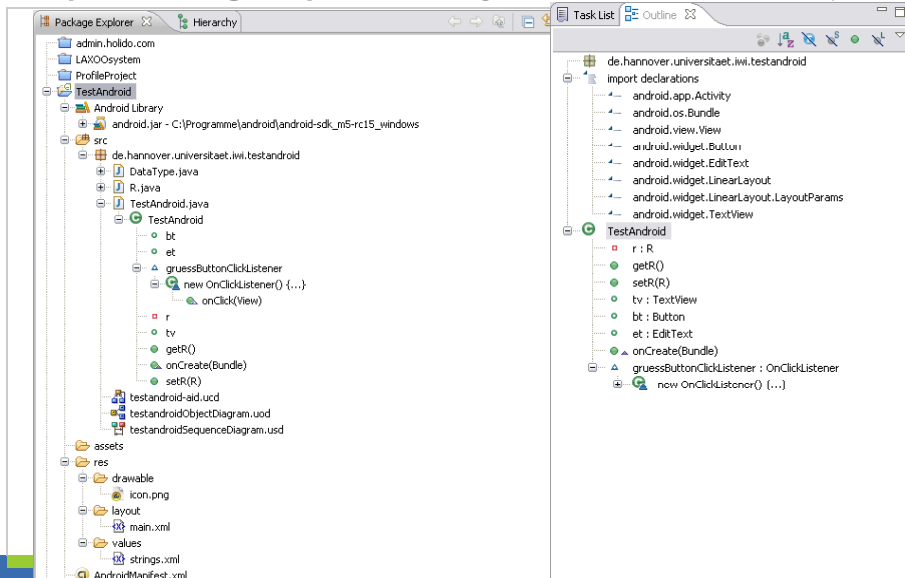
UML Klassenmodell mit Omondo Eclipse UML



UML Sequenzmodellierung mit Omondo Eclipse UML

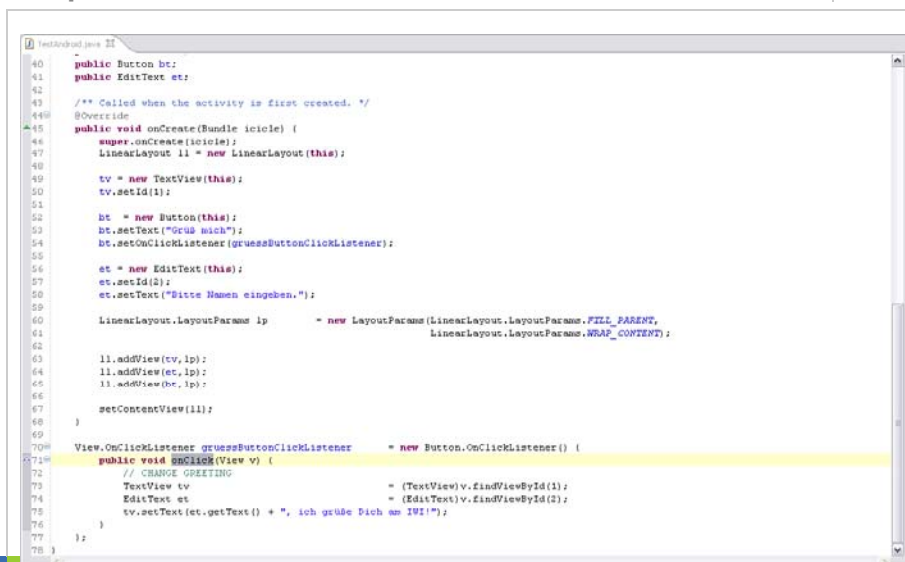


Programmierung des Prototypen mit Eclipse: Package Explorer & Project Outline



The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer displays the project structure for 'TestAndroid', including the 'src' folder with 'TestAndroid.java' and 'R.java', and the 'res' folder with 'layout/main.xml' and 'values/strings.xml'. On the right, the Project Outline view shows the class hierarchy for 'TestAndroid', listing fields like 'r', 'tv', 'bt', 'et' and methods like 'getR()', 'setR()', 'onCreate(Bundle)', and 'new OnClickListener() (...)'. The bottom status bar indicates the date '19.05.2009' and slide number '# 57'.

Programmierung des Prototypen mit Eclipse: Der Java Editor



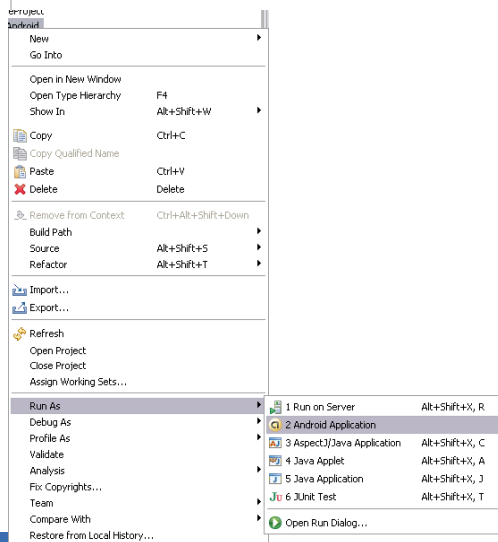
The screenshot shows the Eclipse IDE's Java Editor with the source code of 'TestAndroid.java'. The code includes field declarations for 'Button bt' and 'EditText et', an 'onCreate' method that initializes the UI with a 'LinearLayout', and an 'OnClickListener' implementation for the button. The code is as follows:

```

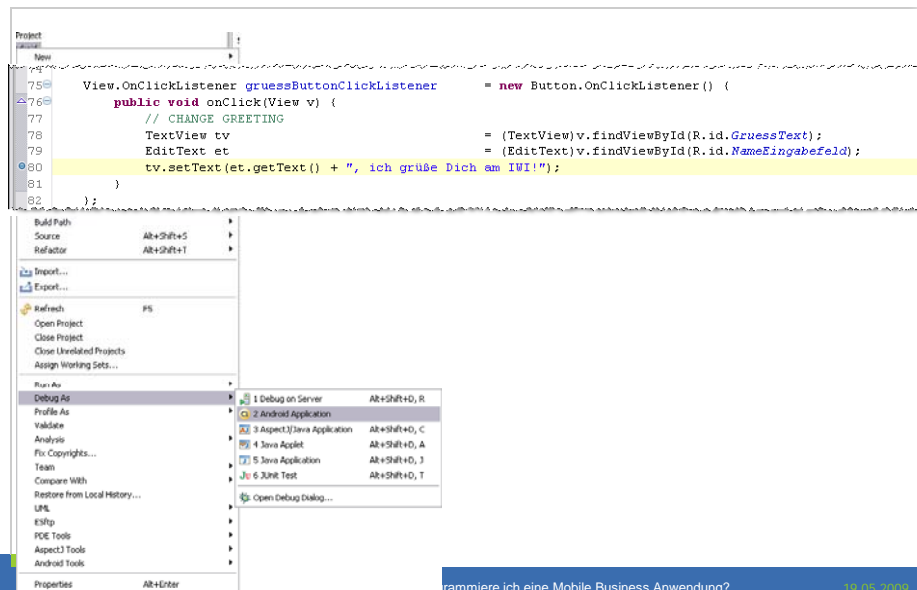
40 public Button bt;
41 public EditText et;
42
43 /** Called when the activity is first created. */
44 @Override
45 public void onCreate(Bundle savedInstanceState) {
46     super.onCreate(savedInstanceState);
47     LinearLayout ll = new LinearLayout(this);
48
49     tv = new TextView(this);
50     tv.setId(1);
51
52     bt = new Button(this);
53     bt.setText("Gruß maske");
54     bt.setOnClickListener(gruessButtonClickListener);
55
56     et = new EditText(this);
57     et.setId(2);
58     et.setText("Bitte Namen eingeben.");
59
60     LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
61         LinearLayout.LayoutParams.FILL_PARENT,
62         LinearLayout.LayoutParams.WRAP_CONTENT);
63
64     ll.addView(tv, lp);
65     ll.addView(et, lp);
66     ll.addView(bt, lp);
67
68     setContentView(ll);
69
70     OnClickListener gruessButtonClickListener = new Button.OnClickListener() {
71
72         public void onClick(View v) {
73             // CHANGE GREETING
74             TextView tv = (TextView)v.findViewById(1);
75             EditText et = (EditText)v.findViewById(2);
76             tv.setText(et.getText() + ", ich grüße Dich am IWI!");
77         }
78     };

```

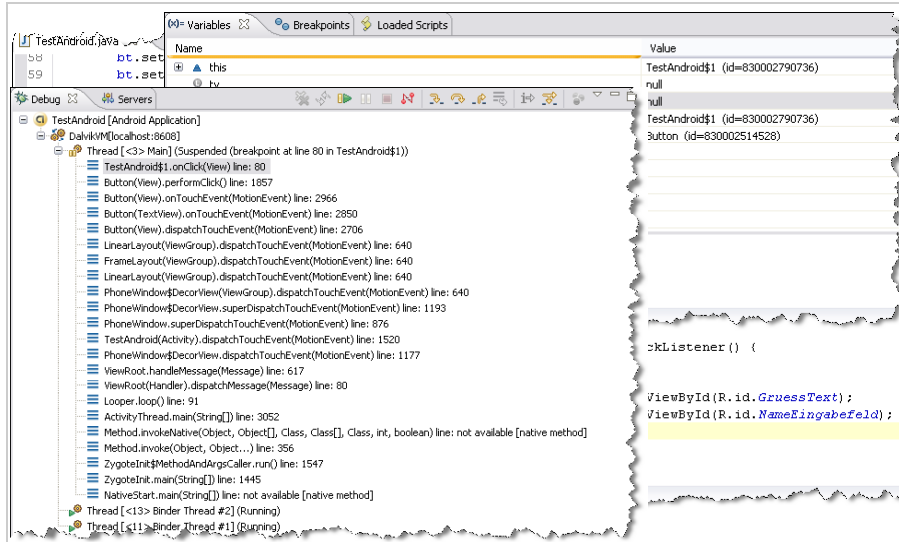
Programmierung des Prototypen mit Eclipse: Start im Android-Emulator



Programmtest und Fehlersuche mit Eclipse: Debugging



Programmtest und Fehlersuche mit Eclipse: Debugging (2)



Variables

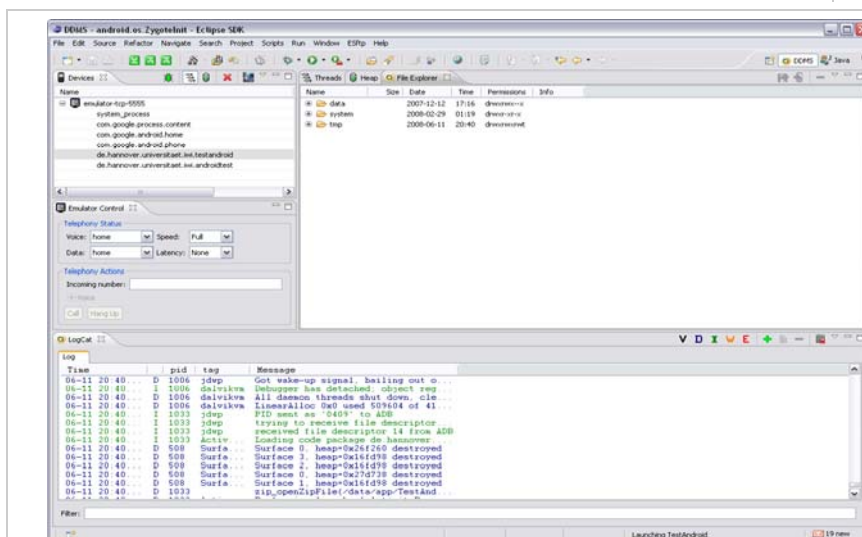
Name	Value
this	TestAndroid#1 (id=830002790736)
bt	null
TestAndroid#1 (id=830002790736)	null
TestAndroid#1 (id=830002790736)	TestAndroid#1 (id=830002790736)
button (id=830002514526)	

Debug Console

```

TestAndroid [Android Application]
  DalvikVM(Localhost:8608)
    Thread [C3> Main] (Suspended (breakpoint at line 80 in TestAndroid#1))
      TestAndroid#1.onClick(View) line: 80
      Button(View).performClick() line: 1857
      Button(View).onTouchEvent(MotionEvent) line: 2966
      Button(TextView).onTouchEvent(MotionEvent) line: 2850
      Button(View).dispatchTouchEvent(MotionEvent) line: 2706
      LinearLayout(ViewGroup).dispatchTouchEvent(MotionEvent) line: 640
      FrameLayout(ViewGroup).dispatchTouchEvent(MotionEvent) line: 640
      LinearLayout(ViewGroup).dispatchTouchEvent(MotionEvent) line: 640
      PhoneWindow$DecorView(ViewGroup).dispatchTouchEvent(MotionEvent) line: 640
      PhoneWindow$DecorView.superDispatchTouchEvent(MotionEvent) line: 1193
      PhoneWindow.superDispatchTouchEvent(MotionEvent) line: 876
      TestAndroid(Activity).dispatchTouchEvent(MotionEvent) line: 1520
      PhoneWindow$DecorView.dispatchTouchEvent(MotionEvent) line: 1177
      ViewRoot.handleMessage(Message) line: 617
      ViewRoot(Handler).dispatchMessage(Message) line: 80
      Looper.loop() line: 91
      ActivityThread.main(String[]) line: 3052
      Method.invokeNative(Object, Object[], Class, int, boolean) line: not available [native method]
      Method.invoke(Object, Object...) line: 356
      ZygooteInit$MethodAndArgsCaller.run() line: 1547
      ZygooteInit.main(String[]) line: 1445
      NativeStart.main(String[]) line: not available [native method]
    Thread [<1> Binder Thread #2] (Running)
    Thread [<11> Binder Thread #1] (Running)
    
```

Kontrolle des Emulators



DDMS - android.os.ZygooteInit - Eclipse SDK

Name	Size	Date	Time	Permissions	Info
data	2007-12-13	17:16	drwxrwxr-x		
system	2008-02-29	01:19	drwxr-xr-x		
tmp	2008-06-11	20:40	drwxrwxrwt		

LogCat

```

Time      pid    tag      Message
06-11 20:40:00.000 D 1006 jdpdp Got wake-up signal; bailing out o...
06-11 20:40:00.000 I 1006 dalvikvm Debugger has detached; object reg...
06-11 20:40:00.000 D 1006 dalvikvm All daemon threads shut down; cle...
06-11 20:40:00.000 D 1006 dalvikvm LinearAlloc (obj) used 509164 of 41...
06-11 20:40:00.000 I 1033 jdpdp PID smnt as "0409" to ADB
06-11 20:40:00.000 I 1033 jdpdp trying to receive file descriptor...
06-11 20:40:00.000 I 1033 jdpdp received file descriptor 14 from ADB...
06-11 20:40:00.000 I 1033 actiiv Loading code package de.hannover...
06-11 20:40:00.000 D 509 Surface Surface 0 heap=0x261260 destroyed
06-11 20:40:00.000 D 509 Surface Surface 1 heap=0x161d78 destroyed
06-11 20:40:00.000 D 509 Surface Surface 2 heap=0x161d78 destroyed
06-11 20:40:00.000 D 509 Surface Surface 3 heap=0x161d78 destroyed
06-11 20:40:00.000 D 509 Surface Surface 4 heap=0x161d78 destroyed
06-11 20:40:00.000 D 1033 zip_openZipFile(/data/app/TestAnd...
    
```

Ausblick



- Modellgetriebene Entwicklung mit Eclipse,
- Rapid Prototyping mit Eclipse (EMF),
- Kontextbezogene Entwicklung (Mylyn),
- Integration von C# in Eclipse (Mono)?,
- Integration von Bugtracking, Mylyn und Eclipse,
- bessere Unterstützung von Sprachen jenseits von Java (Eclipse 4.0),
- Annäherung von Eclipse und Microsoft?.