

Project Flip 20

Studienarbeit

Anhang A

Begriffs- & Abkürzungsverzeichnis
Literaturverzeichnis und Referenzen

Elmer Lukas, Heidt Christina, Treichler Delia
22. Dezember 2011

1 Glossar

1.1 Begriffserklärung

Begriff	Beschreibung
.NET	Software-Plattform der Microsoft Corporation
Accessibility	Zugänglichkeit. Gegenstände und Medien werden so gestaltet, dass sie von jedem Menschen uneingeschränkt genutzt werden können, unabhängig von einer allfälligen Behinderung.
Assembly	In Assemblies werden übersetzte Klassen als ausführbare Dateien bereitgestellt.
Code Behind	Dabei handelt es sich um die CS-Datei, welche zu einer View (.xaml) Datei gehört. Darin enthaltene Funktionen können nur schwer oder gar nicht getestet werden.
Creative Workshop	Workshop, mit welchem Ideen gesammelt werden
Eager Loading	Verbundene Daten werden geladen bevor sie gebraucht werden
Flyweight	Zu Deutsch Fliegengewicht. Entwurfsmuster, das angewendet wird, wenn eine grosse Anzahl von Objekten verwendet wird, welche die gleichen Informationen beinhalten.
Graphical User Interface	Die grafische Benutzeroberfläche ermöglicht dem Benutzer die Interaktion mit dem Computer über grafische Symbole.
Milestone	Zwischenziel eines Projektes
multithreading safe	Möglichkeit, Prozesse ohne Konflikte parallel abzuwickeln
Namespace	Zu Deutsch Namensraum. Struktur für Objekte, diese können über eindeutigen Pfadnamen angesprochen werden
Overhead	Mehraufwand
Papierprototyp	Zeigt das Design einer zu erstellenden Software auf Papier und wird zum Testen der Software erstellt
Perspective Wall	Visualisierungstechnik um eine grosse Anzahl an Informationen darzustellen
Project Note	Eine Beschreibung eines bestimmten Projektes auf einer A4-Seite, welche Aufschluss über die Projektaufgabe, dessen Umsetzung, den Projektpartner und den Kundennutzen des Projektes gibt.
Rastergrafik	Eine Rastergrafik besteht aus einer festen Anordnung von Bildpunkten, welchen je eine Farbe zugeordnet ist. Daher hat eine Rastergrafik auch eine feste Bildgrösse.
Redmine	Webbasiertes Projektmanagement-Tool auf der Basis von Ruby on Rails
RUP	Vorgehensmodell zur Entwicklung von Software
Scrum	Vorgehensmodell zur Entwicklung von Software
Sharepoint	Serverprodukt der Microsoft Corporation
Single tab	Kurzes, einmaliges Antippen
Surface	Ein Computer in Form einer Tischplatte der Microsoft Corporation, bei dem sämtliche Eingaben mit der Hand gemacht werden (Touch-Tisch)
Swipe	Wischbewegung
Tablet-PC	Ein tragbarer Computer, welcher per Eingabestift oder Finger bedient werden kann.
Test Driven Development	Methode, bei welcher zuerst die Tests und dann die zu testenden Komponenten implementiert werden.
ThreadPool	Stellt Arbeitsaufgaben bereit, verarbeitet asynchrone Ein-/Ausgabe, wartet im Auftrag anderer Threads und verarbeitet Zeitgeber.
User Story	Softwareanforderung in 1-2 Sätzen formuliert. Werden in der agilen Softwareentwicklung eingesetzt
Vektorgrafik	Eine Vektorgrafik wird durch Linien/Kurven, Linienstärken und Farben beschrieben. Vektorgrafiken sind beliebig skalierbar.
XPS	Ein Dateiformat der Microsoft Corporation

1.2 Abkürzungserläuterung


Begriff	Beschreibung
GUI	Graphical User Interface
MVVM	Model View ViewModel
PF2	Project Flip 2.0
PN	Project Note
RUP	Rational Unified Process
SDK	Software Development Kit
TDD	Test Driven Development
WPF	Windows Presentation Foundation
XPS	XML Paper Specification

1 Dokumentinformationen

Datum	Version	Änderung	Autor
10.10.2011	1.0	Erste Version des Dokuments	cheidt
17.10.2011	1.1	Verzeichnis ergänzt	cheidt
04.11.2011	1.2	Verzeichnis ergänzt	cheidt
15.11.2011	1.3	Verzeichnis ergänzt	lelmer
01.12.2011	1.4	Verzeichnis ergänzt	lelmer
22.12.2011	1.5	Verzeichnis ergänzt , Review	lelmer

2 Literaturverzeichnis

- [eilbrecht07] Karl Eilbrecht, Gernot Starkte, "Patterns kompakt, Entwurfsmuster für effective Software-Entwicklung", 2. Auflage, Spektrum Verlag, ISBN-13: 978-3-8274-1591-2, 2007
- [mackinlay91] Jock D. Mackinlay, George G. Robertson, Stuart K. Card, ACM, „CHI '91 Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology“, <http://dl.acm.org/citation.cfm?id=108870>
letzter Zugriff: 20.12.2011
- [marx10] Steve Marx, Windows Azure Team, "Pivot View of Netflix Instant Watch Movies", Beispiel eines Pivot Viewers mit einer Filmbibliothek, 30. Juni 2010
<http://netflixpivot.cloudapp.net/>
letzter Zugriff: 22.12.2011
- [microsoft11] Microsoft Corporation, „Microsoft Surface 2.0 Design and Interaction Guide“, Juli 2011, <http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=26713>,
letzter Zugriff: 20.12.2011
- [microsoft11.2] Microsoft Corporation, „Microsoft Code Metrics Values“, November 2011, <http://msdn.microsoft.com/en-us/library/bb385914.aspx>
letzter Zugriff: 20.12.2011
- [shen06] Shen, C.; Ryall, K.; Forlines, C.; Esenther, A.; Vernier, F.D.; Everitt, K.; Wu, M.; Wigdor, D.; Morris, M.R.; Hancock, M.; Tse, E.; , "Informing the Design of Direct-Touch Tabletops," Computer Graphics and Applications, IEEE , vol.26, no.5, pp.36-46, doi: 10.1109/MCG.2006.109, September bis Oktober 2006
letzter Zugriff: 22.12.2011
<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1683692&isnumber=35448>

The perspective wall: detail and context smoothly integratedFull Text:  [pdf](#)

Authors: [Jock D. Mackinlay](#) [Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA](#)
[George G. Robertson](#) [Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA](#)
[Stuart K. Card](#) [Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA](#)

Published in:

- Proceeding
CHI '91 Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology

[ACM](#) New York, NY, USA ©1991[table of contents](#) ISBN:0-89791-383-3 doi>[10.1145/108844.108870](#)

1991 Article

**Bibliometrics**

- Downloads (6 Weeks): 16
- Downloads (12 Months): 175
- Citation Count: 196

Tools and Resources[Request Permissions](#)

TOC Service:

[Email](#) [RSS](#)[Save to Binder](#)

Export Formats:

[BibTeX](#) [EndNote](#) [ACM Ref](#)

Upcoming Conference:

[CHI '12](#)

Share:



Tags: [animation](#) [design](#) [human factors](#) [human factors](#) [interaction](#) [styles](#) [theory](#)

Powered by **THE ACM GUIDE TO COMPUTING LITERATURE**

The ACM Digital Library is published by the Association for Computing Machinery. Copyright © 2011 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

Pivot View of Netflix Instant Watch Movies

Netflix Instant Watch Results

Sort: Rating ▾

Search...

Rating



Genre

MPAA Rating

Year

Cast

Director

AvailableFrom

AvailableTo



delivered by

NETFLIX

Created by Steve Marx; [Windows Azure](#)

[Learn how it was built.](#)



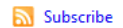
Powered by
Windows Azure™

1 of 1

20.12.2011 13:53



Microsoft® Surface® 2 Design and Interaction Guide



Quick links

- ↓ [Overview](#)
- ↓ [System requirements](#)
- ↓ [Instructions](#)

Microsoft® Surface® 2 Design and Interaction Guide

Quick details

Version:	2.0	Date Published:	7/11/2011
Language:	English		
File Name		Size	
Microsoft Surface 2.0 Design and Interaction Guide.pdf		2.8 MB	DOWNLOAD

Overview

The Microsoft Surface 2.0 Design and Interaction Guide helps designers and developers create Surface applications for Microsoft Surface and Windows 7 touch PCs. Developing compelling Surface experiences requires a different approach to interface design. This document presents design principles and guidelines to address key aspects of application interface design including: interaction, visual, sound, text, and more. These principles and practices are a starting point to get the most out of the Surface software and hardware platform's unique capabilities.

[↑ Top of page](#)

System requirements

Supported Operating Systems: Windows 7

The file is in .PDF format, so a .PDF reader is required.

[↑ Top of page](#)

Instructions

Download the document and open it with a compatible reader.

[↑ Top of page](#)

Code Metrics Values



Visual Studio 2010

Code metrics is a set of software measures that provide developers better insight into the code they are developing. By taking advantage of code metrics, developers can understand which types and/or methods should be reworked or more thoroughly tested. Development teams can identify potential risks, understand the current state of a project, and track progress during software development.

Software Measurements

The following list shows the code metrics results that Visual Studio calculates:

- **Maintainability Index** – Calculates an index value between 0 and 100 that represents the relative ease of maintaining the code. A high value means better maintainability. Color coded ratings can be used to quickly identify trouble spots in your code. A green rating is between 20 and 100 and indicates that the code has good maintainability. A yellow rating is between 10 and 19 and indicates that the code is moderately maintainable. A red rating is a rating between 0 and 9 and indicates low maintainability.
- **Cyclomatic Complexity** – Measures the structural complexity of the code. It is created by calculating the number of different code paths in the flow of the program. A program that has complex control flow will require more tests to achieve good code coverage and will be less maintainable.

Note

In some cases, the calculation of the cyclomatic complexity for a method in Visual Studio 2010 differs from earlier versions. For more information, see the "Changes in Visual Studio 2010 code complexity calculations section" of [Troubleshooting Code Metrics Issues](#).

- **Depth of Inheritance** – Indicates the number of class definitions that extend to the root of the class hierarchy. The deeper the hierarchy the more difficult it might be to understand where particular methods and fields are defined or/and redefined.
- **Class Coupling** – Measures the coupling to unique classes through parameters, local variables, return types, method calls, generic or template instantiations, base classes, interface implementations, fields defined on external types, and attribute decoration. Good software design dictates that types and methods should have high cohesion and low coupling. High coupling indicates a design that is difficult to reuse and maintain because of its many interdependencies on other types.
- **Lines of Code** – Indicates the approximate number of lines in the code. The count is based on the IL code and is therefore not the exact number of lines in the source code file. A very high count might indicate that a type or method is trying to do too much work and should be split up. It might also indicate that the type or method might be hard to maintain.

Anonymous Methods

An *anonymous method* is just a method that has no name. Anonymous methods are most frequently used to pass a code block as a delegate parameter. Metrics results for an anonymous method that is declared in a member, such as a method or accessor, are associated with the member that declares the method. They are not associated with the member that calls the method.

For more information about how Code Metrics treats anonymous methods, see [Anonymous](#)

[Methods and Code Analysis.](#)

Generated Code

Some software tools and compilers generate code that is added to a project and that the project developer either does not see or should not change. Mostly, Code Metrics ignores generated code when it calculates the metrics values. This enables the metrics values to reflect what the developer can see and change.

Code generated for Windows forms is not ignored, because it is code that the developer can see and change.

See Also

Other Resources

[Measuring Complexity and Maintainability of Managed Code](#)

Community Content

© 2011 Microsoft. All rights reserved.

IEEE Xplore®
DIGITAL LIBRARY

SEARCH

Advanced Search | Preferences | Search Tips

BROWSE ▼

MY SETTINGS ▼

CART

SIGN OUT

About IEEE Xplore | Terms of Use | Feedback

Help

ON THIS PAGE

Abstract

Index Terms

References


Cited by IEEE

Browse > Journals > Computer Graphics and Applicat ... > Volume: 26 Issue: 5


Informing the Design of Direct-Touch Tabletops

 Access Full Text

 Download Citation

 Email

 Print

 Request Permissions

Shen, C.; Ryall, K.; Forlines, C.; Esenther, A.; Vernier, F.D.; Everitt, K.; Wu, M.; Wigdor, D.; Morris, M.R.; Hancock, M.; Tse, E.; Mitsubishi Electr. Res. Labs.

This paper appears in: [Computer Graphics and Applications, IEEE](#)

Issue Date: Sept.-Oct. 2006

Volume: 26 Issue: 5

On page(s): 36 - 46

ISSN: 0272-1716

References Cited: 18

Cited by : 4

INSPEC Accession Number: 9089936

Digital Object Identifier: [10.1109/MCG.2006.109](#)

Date of Current Version: 28 August 2006

Sponsored by: [IEEE Computer Society](#)

ABSTRACT

Tables provide a large and natural interface for supporting direct manipulation of visual content for human-to-human interactions. Such surfaces also support collaboration, coordination, and parallel problem solving. However, the direct-touch table metaphor also presents considerable challenges, including the need for input methods that transcend traditional mouse- and keyboard-based designs. In this paper, we've designed, implemented, and studied a variety of tabletop user interfaces, interaction techniques, and usage scenarios

BROUGHT TO YOU BY

FHO East

Your institute subscribes to:

IEEE/IET Electronic Library (IEL), IBM Journal of Research and Development, VDE VERLAG Conference Proceedings

[What can I access?](#)

[Terms of Use](#)