

Übung zur Vorlesung  
**Digitale Medien**

Hanna Schneider  
Ludwig-Maximilians-Universität München  
Wintersemester 2014/2015

# Good to Know

- Follow [@DM\\_WS1415](#) (Folien oder Übungen hochgeladen, Erinnerungen etc.)
- Informatiker Forum  
<http://www.die-informatiker.net/>
- Mimuc Twitter Account (inoffiziell)  
<http://twitter.com/mimuc>
- Medieninformatik LMU Facebook Gruppe (inoffiziell)  
<https://www.facebook.com/groups/36775131102/>

# Übungsbetrieb

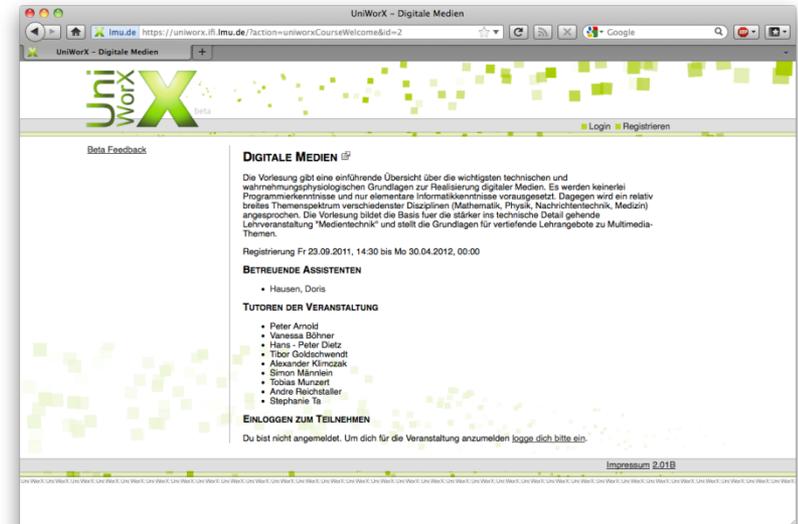
- Informationen zu den Übungen:  
<http://www.medien.ifi.lmu.de/dm>
- Praktische Anwendungen des theoretischen Vorlesungsstoffs
- Wichtige Voraussetzung für die Klausur
- Bonuspunkte für die Klausur

# UniWorX

- UniWorX:  
<https://uniworx.ifi.lmu.de>
- Anmeldung ist Voraussetzung für Übungsteilnahme
- Anmeldung für die Übungsgruppen:  
seit Freitag, 10.10., bis Freitag 24.10

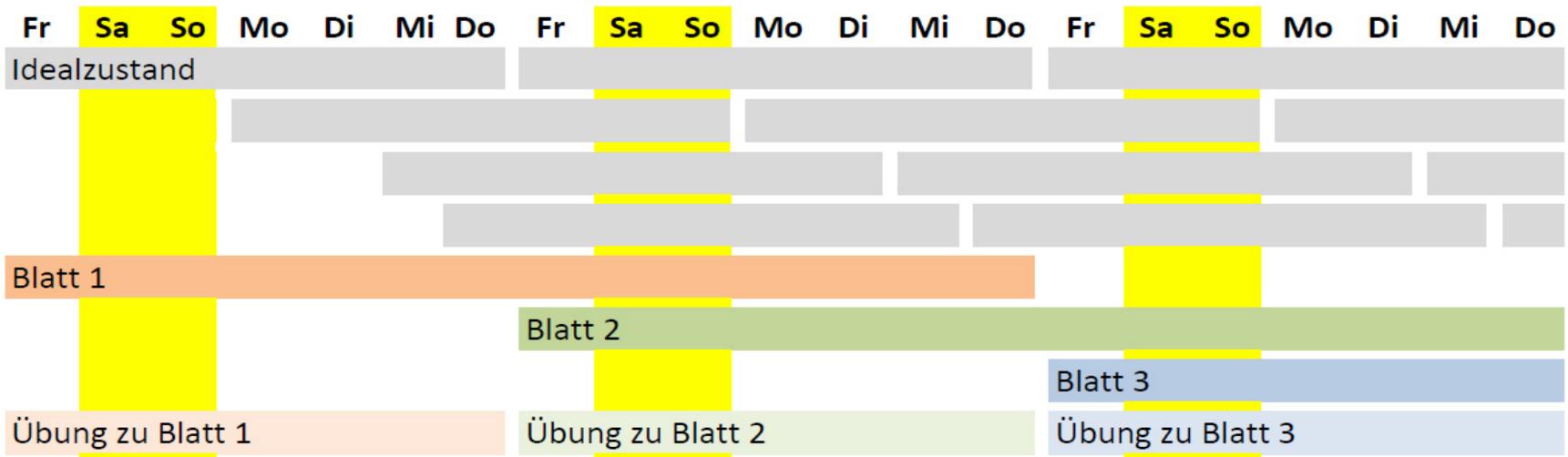
Hinweise:

- Benötigt Campus Kennung:
- Mails von UniWorX werden an diese E-Mail versendet  
Regelmäßig Adresse checken oder weiterleiten!

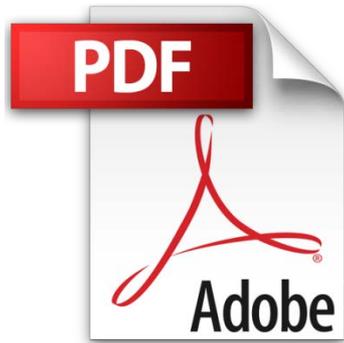


# Übungsblätter

- Ausgabe:
  - jeden Freitag ab (spätestens) 14 Uhr
  - auf <http://www.medien.ifi.lmu.de/dm/>
- Abgabe:
  - jeweils spätestens übernächster Freitag 9 Uhr (morgens!)
  - über <https://uniworx.ifi.lmu.de>
  - Achtung: erneutes Hochladen überschreibt die alte Lösung



# Abgabeformate



0 Punkte für Abgaben in anderen Formaten!

# Benotung

- Klausur am Ende des Semesters  
(4.02.14, 10:00 – 12:00)
  - Nachholklausur am Ende der Semesterferien
- Abgabe der Übungsblätter:
  - freiwillig
  - ABER: dringend empfohlen
- Anreiz:  
Bonuspunkte aus dem Übungsbetrieb

# Bonuspunkte

- Pro Übungsblatt 20 Punkte
- Übungsblatt bestanden mit 75 % → 15 Punkte
- Bestandene Übungsblätter gehen in Bonuswertung ein
  
- Maximaler Notenbonus in der Klausur: 15%

# Plagiate

- Abschreiben
  - von Kommilitonen
  - von anderen Quellen (z.B. Wikipedia)Ist in **keiner Weise** erlaubt!
- Wenn Sie erwischt werden, wird:
  - Beim ersten mal das ganze Übungsblatt mit 0 Punkten bewertet
  - Bei einem erneuten Plagiat alle Übungsblätter mit 0 Punkten bewertet (auch die schon bestandenen)
- Im Zweifelsfall immer vor der Abgabe nachfragen.
- Wir prüfen nach – eventuell auch erst nach Ende der Übungen.
- Erlaubt ist:
  - Zusammen lernen
  - Links zu guten Quellen austauschen
  - Gemeinsames Code Review
  - Tricks verraten



# Ansprechpartner

- Gegenseitiger Austausch – auch über das Forum – ist sehr gewünscht.
- Individuelle Probleme (z.B. mit der Korrektur) lassen sich aber meist deutlich besser persönlich klären. Daher bitte direkt den betroffenen Tutor oder wenn das nicht zur Klärung führt die Übungsleitung ansprechen.

# **Blatt 1**

## **(mit Bonuspunkte)**

Zahlensysteme  
ASCII

1 Bit		<i>1 / 0, Strom an / Strom aus, ja / nein</i>
1 Byte	= 8 Bit	<i>Ein Buchstabe</i>
1 Kilobyte	= 1.000 Byte	<i>1.000 Buchstaben, 3 Paragraphen Text</i>
1 Megabyte	= 1.000 Kilobyte	<i>Ein Foto, eine Minute Audio (MP3)</i>
1 Gigabyte	= 1.000 Megabyte	<i>2 Stunden komprimierter Film</i>
1 Terabyte	= 1.000 Gigabyte	<i>3 Monate komprimierter Film</i>
1 Petabyte	= 1.000 Terabyte	<i>230 Jahre komprimierter Film</i>

## Umrechnung:

**Bit -> Byte:** durch 8 teilen  
**Byte -> Bit:** mit 8 multiplizieren

**Byte -> Kilobyte:** durch 1.000 teilen  
**Kilobyte -> Byte:** mit 1.000 multiplizieren

# ASCII Code

1 Byte pro Zeichen (1 Byte = 8 Bit)

Beispiel:

Nonsense

Dezimal: 78 111 110 115 101 110 115 101  
 N o n s e n s e

Binär: 01001110 N  
 01101111 o  
 01101110 n  
 01110011 s  
 01100101 e  
 01101110 n  
 01110011 s  
 01100101 e

## 8 Byte

Scan-code	ASCII hex	ASCII dez	Zeichen	Scan-code	ASCII hex	ASCII dez	Zch.	Scan-code	ASCII hex	ASCII dez	Zch.	Scan-code	ASCII hex	ASCII dez	Zch.
	00	0	NUL ^@		20	32	SP		40	64	@	0D	60	96	`
	01	1	SOH ^A	02	21	33	!	1E	41	65	A	1E	61	97	a
	02	2	STX ^B	03	22	34	"	30	42	66	B	30	62	98	b
	03	3	ETX ^C	29	23	35	#	2E	43	67	C	2E	63	99	c
	04	4	EOT ^D	05	24	36	\$	20	44	68	D	20	64	100	d
	05	5	ENQ ^E	06	25	37	%	12	45	69	E	12	65	101	e
	06	6	ACK ^F	07	26	38	&	21	46	70	F	21	66	102	f
	07	7	BEL ^G	0D	27	39	'	22	47	71	G	22	67	103	g
0E	08	8	BS ^H	09	28	40	(	23	48	72	H	23	68	104	h
0F	09	9	TAB ^I	0A	29	41	)	17	49	73	I	17	69	105	i
	0A	10	LF ^J	1B	2A	42	*	24	4A	74	J	24	6A	106	j
	0B	11	VT ^K	1B	2B	43	+	25	4B	75	K	25	6B	107	k
	0C	12	FF ^L	33	2C	44	,	26	4C	76	L	26	6C	108	l
1C	0D	13	CR ^M	35	2D	45	-	32	4D	77	M	32	6D	109	m
	0E	14	SO ^N	34	2E	46	.	31	4E	78	N	31	6E	110	n
	0F	15	SI ^O	08	2F	47	/	18	4F	79	O	18	6F	111	o
	10	16	DLE ^P	0B	30	48	0	19	50	80	P	19	70	112	p
	11	17	DC1 ^Q	02	31	49	1	10	51	81	Q	10	71	113	q
	12	18	DC2 ^R	03	32	50	2	13	52	82	R	13	72	114	r
	13	19	DC3 ^S	04	33	51	3	1F	53	83	S	1F	73	115	s
	14	20	DC4 ^T	05	34	52	4	14	54	84	T	14	74	116	t
	15	21	NAK ^U	06	35	53	5	16	55	85	U	16	75	117	u
	16	22	SYN ^V	07	36	54	6	2F	56	86	V	2F	76	118	v
	17	23	ETB ^W	08	37	55	7	11	57	87	W	11	77	119	w
	18	24	CAN ^X	09	38	56	8	2D	58	88	X	2D	78	120	x
	19	25	EM ^Y	0A	39	57	9	2C	59	89	Y	2C	79	121	y
	1A	26	SUB ^Z	34	3A	58	:	15	5A	90	Z	15	7A	122	z
01	1B	27	Esc ^[	33	3B	59	;		5B	91	[		7B	123	{
	1C	28	FS ^\	2B	3C	60	<		5C	92	\		7C	124	
	1D	29	GS ^]	0B	3D	61	=		5D	93	]		7D	125	}
	1E	30	RS ^^	2B	3E	62	>	29	5E	94	^		7E	126	~
	1F	31	US ^_	0C	3F	63	?	35	5F	95	_	53	7F	127	DEL

# Konvertierung: Binär/Dezimal (ganzzahlig)

## Ganze **Binär**zahlen nach **Dezimal**

Arbeite die Ziffern der Zahl von rechts nach links durch.  
Falls eine Ziffer an der Position  $z$  gleich 1 ist (Achtung: Die rechteste Position ist 0 und nicht 1!), rechne  $2$  hoch  $z$  und addiere die Lösung zum Gesamtergebnis.

Beispiel: **1001011** nach Dezimal

$$\begin{aligned} & 1 * 2^0 + 1 * 2^1 + 1 * 2^3 + 1 * 2^6 \\ = & 1 + 2 + 8 + 64 \\ = & \mathbf{75} \end{aligned}$$

## Ganze **Dezimal**zahlen nach **Binär**

Teile die Zahl durch 2.  
Der verbleibende Rest ist die nächste Ziffer (fülle von rechts nach links auf!).  
Höre auf, sobald das Ergebnis 0 wird.

Beispiel: **75** nach Binär

$75 / 2 = 37$	Rest: <b>1</b>
$37 / 2 = 18$	Rest: <b>1</b>
$18 / 2 = 9$	Rest: <b>0</b>
$9 / 2 = 4$	Rest: <b>1</b>
$4 / 2 = 2$	Rest: <b>0</b>
$2 / 2 = 1$	Rest: <b>0</b>
$1 / 2 = 0$	Rest: <b>1</b>



# Konvertierung: Binär/Dezimal (zwischen 0 und 1)

## *Binär nach Dezimal (Komma)*

Arbeite die Ziffern der Zahl hinter dem Komma von **links nach rechts** durch. Falls eine Ziffer an der Position  $z$  gleich 1 ist (Achtung: Die linkeste Position ist diesmal 1!), rechne  $2^{-z}$  und addiere die Lösung zum Gesamtergebnis.

Beispiel: **0,0101** nach Dezimal

$$\begin{aligned} & 1 * 2^{-2} + 1 * 2^{-4} \\ = & 0,25 + 0,0625 \\ = & \mathbf{0,3125} \end{aligned}$$

Hinweis:

$$2^{-x} = 1 / 2^x$$

## *Dezimal nach Binär (Komma)*

Multipliziere die Zahl mit 2.  
Die Zahl vor dem Komma ist die nächste Zahl des Ergebnisses.  
Entferne die Zahl vor dem Komma.  
Wiederhole das Verfahren, bis nichts mehr rechts vom Komma steht oder sich die Ergebnisse wiederholen.

Beispiel: **0,3125** nach Binär

$$\begin{array}{rcl} 0,3125 & * 2 = & \mathbf{0,625} \\ 0,625 & * 2 = & \mathbf{1,25} \\ 0,25 & * 2 = & \mathbf{0,5} \\ 0,5 & * 2 = & \mathbf{1} \end{array} \quad \downarrow$$

=> **0,0101**

# Rechenbeispiele

**Binär nach Dezimal:**

110011

0,111

**Dezimal nach Binär:**

67

0,755

# Rechenbeispiele

## Binär nach Dezimal:

$$\begin{aligned} &110011 \\ &= 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 0 \cdot 2^3 + \\ &1 \cdot 2^4 + 1 \cdot 2^5 \\ &= 1 + 2 + 16 + 32 \\ &= 51 \end{aligned}$$

$$\begin{aligned} &0,111 \\ &= 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} \\ &= 0,50 + 0,25 + 0,125 \\ &= 0,875 \end{aligned}$$

## Dezimal nach Binär:

$$\begin{array}{ll} 67 & \\ 67/2 = 33 & \text{Rest: 1} \\ 33/2 = 16 & \text{Rest: 1} \\ 16/2 = 8 & \text{Rest: 0} \\ 8/2 = 4 & \text{Rest: 0} \\ 4/2 = 2 & \text{Rest: 0} \\ 2/2 = 1 & \text{Rest: 0} \\ 1/2 = 0 & \text{Rest: 1} \\ \rightarrow & 1000011 \end{array}$$

$$\begin{aligned} &0,755 \\ &0,755 \cdot 2 = 1,51 \\ &0,51 \cdot 2 = 1,02 \\ &0,02 \cdot 2 = 0,04 \\ &0,04 \cdot 2 = 0,08 \end{aligned}$$

$$\begin{aligned} &\dots \\ &\rightarrow 0,1100\dots \end{aligned}$$

# Hexadezimaldarstellung (1)

## Hexadezimal nach Dezimal

**A4C**<sub>16</sub> bzw. 0xA4C

$$\mathbf{C} * 16^0 + \mathbf{4} * 16^1 + \mathbf{A} * 16^2 =$$

$$12 * 1 + 4 * 16 + 10 * 256 =$$

$$12 + 64 + 2560 =$$

**2636**<sub>10</sub>

Google

"0xXXXX to decimal"

"XXXXXX to hex"

Hexadecimal	Binary	Decimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

[Hintergrund und Umrechnung](#)

# Hexadezimaldarstellung (2)

## Hexadezimal nach Dezimal

0xFF

0x7F

0x1C0

Hexadecimal	Binary	Decimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

# Hexadezimaldarstellung (3)

Die Dezimalzahl **76528975** wird ins Hexadezimalsystem umgewandelt.

Gehe nach folgendem Verfahren vor:

- (1) Teile die Zahl mit Rest durch 16.
- (2) Der Divisionsrest ist die nächste Ziffer (von rechts nach links). Für Reste  $> 9$  nimm die Buchstaben A, B, C, D, E, F
- (3) Falls der (ganzzahlige) Quotient = 0 ist, bist du fertig, andernfalls nimm den (ganzzahligen) Quotienten als neue Zahl und wiederhole ab (1).

76528975	: 16 =	4783060	Rest: 15	--> Ziffer: F
4783060	: 16 =	298941	Rest: 4	--> Ziffer: 4
298941	: 16 =	18683	Rest: 13	--> Ziffer: D
18683	: 16 =	1167	Rest: 11	--> Ziffer: B
1167	: 16 =	72	Rest: 15	--> Ziffer: F
72	: 16 =	4	Rest: 8	--> Ziffer: 8
4	: 16 =	0	Rest: 4	--> Ziffer: 4



Resultat: **48FBD4F**

# Big-Endian vs. Little-Endian (1)

**Byteweise Ausleserichtung:**

**Big-Endian:** Höchster Wert zuerst

$$234 \Rightarrow 2 * 100 + 3 * 10 + 4 * 1$$

**Little-Endian:** Niedrigster Wert zuerst

$$234 \Rightarrow 2 * 1 + 3 * 10 + 4 * 100$$

**Sprache:**

24 = „twenty-four“ (Englisch – Big-Endian)

24 = „vierundzwanzig“ (Deutsch – Little-Endian)

**Beispiel:**

439041101 als 32-Bit Integer

Binär:

00011010 00101011 00111100 01001101

Hex:

1A 2B 3C 4D

	Big Endian			Little Endian		
Adresse	Hex	Dez	Binär	Hex	Dez	Binär
10000	1A	26	00011010	4D	77	01001101
10001	2B	43	00101011	3C	60	00111100
10002	3C	60	00111100	2B	43	00101011
10003	4D	77	01001101	1A	26	00011010

Beispiel: Wikipedia

# Big-Endian vs. Little-Endian (1)

33 FA CC 00

Big-Endian: 0x33FACC00 => 872 074 240

Little-Endian: 0x00CCFA33 => 13 433 395

00 00 0F 11

Big-Endian: 0x00000F11 => 3 857

Little-Endian: 0x110F0000 => 286 195 712