java.io   
Class PrintStream

[java.lang.Object](http://download.oracle.com/javase/1.4.2/docs/api/java/lang/Object.html)

extended by[java.io.OutputStream](http://download.oracle.com/javase/1.4.2/docs/api/java/io/OutputStream.html)

extended by[java.io.FilterOutputStream](http://download.oracle.com/javase/1.4.2/docs/api/java/io/FilterOutputStream.html)

extended byjava.io.PrintStream

## java.io Class PrintWriter

[java.lang.Object](http://download.oracle.com/javase/1.4.2/docs/api/java/lang/Object.html)

extended by[java.io.Writer](http://download.oracle.com/javase/1.4.2/docs/api/java/io/Writer.html)

extended byjava.io.PrintWriter

### Tip dönüştürme karekterleri

Java’da *String* formatlamayı ***printf()***metodu ile yapabiliyoruz.

*printf()* kullanımında sözdizimi şöyledir:

System.out.printf(format, argümanlar);

Burada format ve argümanların neler olduğunu açıklamalıyız. Format deyimi, çıktının biçemini belirleyen nakıştır (pattern).

*printf()* metodunda Nakışı Belirleyen Sözdizimi:

printf(%[argument\_index$][flags][width][.precision]conversion)

() içindeki terimlerin anlamları şöyledir:

% Dönüşümü başlatan simge

[] Bu parantezlerin içleri isteğe bağlıdır. Gerekli ise yazılır.

argument\_index$ Değişken\_damgası, değişken değeri için yer tutucu; değişkenin çıktıda nereye konuşlanacağını belirtir.

Flags Bayraklar: - , ^ , 0 , #

conversion : *d,o,x,f,g,a,c,s,b,h,n* harfleri dönüşümün ne olacağını belirler. Dönüşümlerin tanımları aşağıdaki listede verilmiştir.

Örnekler:

*System.out.printf(“%d”, 123.6)* nın çıktısı 124 tür. *d* dönüştürücüsü sayıyı decimal tamsayıya dönüştürür.

*System.out.printf(“%f”,1234567.123)* ün çıktısı 1234567,123000 dır.  *f* dönüştürücüsü sayıyı kesirli sayıya dönüştürür. Aksi istenmezse, kesir hanelerinin öntanımlı (dafault) sayısı 6 dır.

int c = 65;

out**.**printf**(** "result is: [%2x] : %3d : %1c\n"**,** c**,** c**,** c **);**

deyiminin çıktısı şudur:

*// prints : result is:\_[41]\_:\_\_65\_:\_A*

Bu deyimi ve çıktıyı irdeleyelim:

*// %2x ilk değişken için 2 harf genişliğinde yer tutar; hex olarak yazar*

*// %3d sonraki değişken için 3 harf genişliğinde yer tutar; decimal olarak yazar*

*// %1c sonraki değişkene 1 harf genişliğinde yer tutar; Unicode olarak yazar*

*// Geri kalanların hepsi [] dahil dekoratiftir*

Tip dönüştürme karekterleri

|  |  |
| --- | --- |
| 'd': decimal integer | (10 tabanlı tamsayı) |
| 'o': octal integer | (8 tabanlı tamsayı) |
| 'x': hexadecimal integer | (16 tabanlı tamsayı) |
|  |  |
| 'f': decimal notation for float | (kesirli sayı göster) |
| 'g': scientific notation (with an exponent) for float | (kesirli sayıyı üstel göster) |
| 'a': hexadecimal with an exponent for float | (16 tabanlı sayıyı üstel göster) |
| 'c': for a character | (karekter göster) |
| 's': for a string. | (stringe dönüştür) |
| 'b': for a boolean value, so its output is "true" or "false". | (mantıksal değere dönüştür) |
| 'h': output the hashcode of the argument in hexadecimal form. | ((hash kodunu hex olarak göster) |
| 'n': "%n" has the same effect as "\n". | (satırbaşı yap) |

                                     
argument\_index:  "1$"  ilk değişkene yet tutar   
                       "2$"  ikinci değişkene yet tutar   
                       '<'  önceki formatın kullanılacağını belirtir.                     

 flags:  '-'  left-justified (sola dayalı yaz)  
              '^'  and uppercase (Büyük harfle yaz)  
              '+'  output a sign **for**numerical values. (pozitif/negatif işareti koy)  
              '0'  forces numerical values to be zero-padded.   (nakışın hanelerini doldur)                                   
        
width:  (genişlik) Çıktının kaç kolona yazılacağını belirler.

precision: (duyarlık) Kesir hanelerinin sayısını belirler.

typechar  dönüşüm tipini belirleyen karekterdir. Geçerli dönüşüm karekterleri aşağıdaki listede gösterilmektedir.

|  |  |  |
| --- | --- | --- |
| type character | input | string result |
| %d | signed int | signed decimal integer [işaretli decimal (10 tabanlı) tamsayı] |
| %u | unsigned int | unsigned decimal integer [işaretsiz decimal (10 tabanlı) tamsayı] |
| %o | unsigned int | unsigned octal integer [işaretsiz octal (8 tabanlı) tamsayı] |
| %x, %X | unsigned int | unsigned hexadecimal integer, lowercase or uppercase [işaretsiz hexadecimal (16 tabanlı) tamsayı] |
| %z[n], %Z[n] | unsigned int | unsigned integer base n, with n coded in decimal; include square brackets |
| %f | float | Kesirli sayı, standart notasyon |
| %e, %E | float | Kesirli sayı, bilimsel notasyon (üstel için e ya da E kullanır) |
| %g, %G | float | Değerine bağlı olarak  %f ya da %e rolünü oynar. Üst -4 ten büyük olduğunda bilimsel notasyon kullanır. |
| %s | String | string |
| %c | char | karekter |
| %p | Object | Object, işaretsiz hexadecimal olarak hash kodu belirtir (örn, pointer değeri) |
| Ek belirleyiciler | | |
| %\n | (none) | platform-bağımsız satır ayracı |
| %n | (null) | Karekterleri sayar |

Tip dönüşüm karekterleri yalnızca çıktının değil, girdinin de tipini belirler.

**Örnekler**

package JPrintf;

public class Jprintf01 {

public static void main(String args[]) {

System.out.printf("%d %(d %+d %05d\n", 3, -3, 3, 3);

System.out.printf("Default floating-point format: %f \n", 1234567.123);

System.out.printf("Floating-point with commas: %,f \n", 1234567.123);

System.out.printf("Negative floating-point default: %,f \n", -1234567.123);

System.out.printf("Negative floating-point option: %,(f \n", -1234567.123);

System.out.printf("Line-up positive and negative values: \n");

System.out.printf("% ,.2f \n% ,.2f \n", 1234567.123, -1234567.123);

}

}

package JPrintf;

public class Jprintf02 {

static int a, b, c;

static double x, y, z;

public static void main(String[] args) {

x = 27.5;

y = 33.75;

System.out.printf("x = %f y = %g", x, y);

System.out.println();

a = 5;

b = 15;

c = 255;

System.out.printf("a = %d b = %x c = %o", a, b, c);

System.out.println();

x = 27.5;

y = 33.75;

System.out.printf("x = %2$f y = %1$g", x, y);

System.out.println();

// a = 5; b = 15; c = 255;

System.out.printf("a = %3$d b = %1$x c = %2$o", a, b, c);

}

}

/\*

x = 27,500000 y = 33.7500

a = 5 b = f c = 377

x = 33,750000 y = 27.5000

a = 255 b = 5 c = 17

\*/

package JPrintf;

public class Jprintf03 {

static int a, b, c;

static double x, y, z;

public static void main(String[] args) {

double x = 47.8;

y = 55.67;

System.out.printf("x = %15f y = %8g", x, y);

System.out.println();

int a = 3;

b = 26;

c = 354;

System.out.printf("a = %1$5d b = %2$5x c = %3$2o", a, b, c);

System.out.println();

a = 11;

b = 121;

c = 251;

System.out.printf("%na = %1$-5d b = %2$-5x c = %3$-5o", a, b, c);

System.out.println();

a = 25;

b = 55;

c = 155;

System.out.printf("a = %3$d b = %1$x c = %2$o", a, b, c);

}

}

/\*

x = 47,800000 y = 55.6700

a = 3 b = 1a c = 542

a = 11 b = 79 c = 373

a = 155 b = 19 c = 67

\*/

package JPrintf;

public class Jprintf04 {

public static void main(String[] args) {

int count = 0;

for (int ch = 'a'; ch <= 'z'; ch++) {

System.out.printf(" %1$4c%1$4x", ch);

if (++count % 6 == 0) {

System.out.printf("%n");

}

System.out.printf(" %1$4c%<4x", ch);

}

}

}

/\*

a 61 a 61 b 62 b 62 c 63 c 63 d 64 d 64 e 65 e 65 f 66

f 66 g 67 g 67 h 68 h 68 i 69 i 69 j 6a j 6a k 6b k 6b l 6c

l 6c m 6d m 6d n 6e n 6e o 6f o 6f p 70 p 70 q 71 q 71 r 72

r 72 s 73 s 73 t 74 t 74 u 75 u 75 v 76 v 76 w 77 w 77 x 78

x 78 y 79 y 79 z 7a z 7a

\*/

**public** **class** Jprintf05 {

**public** **static** **void** main(String[] a) {

**int** yüzde = 85;

System.*out*.printf("\n%1$d%%", yüzde);

String str = "Çevik kurnaz tilki.";

System.*out*.printf("%nThe string is:%n%s%n%1$25s", str);

}

}

/\*

85%

The string is:

Çevik kurnaz tilki.

Çevik kurnaz tilki.

\*/

*Açıklamalar:*

4-ünci satırda % simgesini yazdırmak için %% yazılmıştır.

Son deyimdekilerin anlamları şöyledir:

% : dönüşümün başlayacağını belirtir

*%n* : harfi daha önceki formatta geçen \n yerine geçer.

%s : "Çevik kurnaz tilki." stringini aynen yazar

%1$ ilk argüman (*str*) için yer tutucudur. Çıktıda *str* burauya yazılacaktır.

1$25s : argümanın 25 kolona sağa yanaşık bir string olarak yazılacağını belirtir.

**public** **class** Jprintf06 {

**public** **static** **void** main(String[] args) {

**int** n = 123, m = 496;

**double** x = 38.7, y = 49.83;

String strA = String.*format*("n = %15d m = %14o", n, m);

String strB = String.*format*("n = %15x m = %14h", n, m);

String str1 = String.*format*("x = %15.0f y = %14.3g", x, y);

String str2 = String.*format*("x = %15.2f y = %14.3E", x, y);

String str3 = String.*format*("x = %15.2a y = %14.3e", x, y);

System.*out*.println(strA);

System.*out*.println(strB);

System.*out*.println(str1);

System.*out*.println(str2);

System.*out*.println(str3);

System.out.printf( "%s\n%s\n", "Java programlamaya", "Hoş geldiniz!" );

}

}

/\*

n = 123 m = 760

n = 7b m = 1f0

x = 39 y = 49.8

x = 38,70 y = 4.983E+01

x = 0x1.36p5 y = 4.983e+01

Java programlamaya

Hoş geldiniz!

\*/

public class Jprintf07 {

public static void main(String[] a) {

StringBuffer buf = new StringBuffer();

java.util.Formatter formatter = new java.util.Formatter(buf);

double x = 27.5, y = 33.75;

formatter.format("x = %15.2f y = %14.3g", x, y);

System.out.print(buf);

}

}

**public** **class** Jprintf08 {

**static** **int** *c* = 68;

**public** **static** **void** main(String[] args) {

System.*out*.printf("Çıktı: [%2x] : %3d : %1c\n", *c*, *c*, *c*);

System.out.printf("Çıktı: %4.3g\n", 1.4d); // Çıktı: 1.40

System.out.printf("Çıktı: %4.3g\n", 1.456d); // Çıktı: 1.46

System.out.printf("Çıktı: %4.3g\n", 145.0d); // Çıktı: 145

System.out.printf("Çıktı: %8.4g\n", 1.5d); // Çıktı: 1.500

System.out.printf("Çıktı: %4.4g\n", 1.5E-6); // Çıktı: 1.500e-06

System.out.printf("Çıktı: %,8.6g\n", 12345.0d); // Çıktı: 12,345.0

}

}

/\*

Çıktı: [44] : 68 : D

Çıktı: 1.40

Çıktı: 1.46

Çıktı: 145

Çıktı: 1.500

Çıktı: 1.500e-06

Çıktı: 12,345.0

\*/

*// This works only in JDK 1.5+*

import static java**.**lang**.**System**.**out**;**

*// ...*

int **c =** 65**;**

*// printf works much like a Fortran FORMAT or C printf.*

*// %2x means insert first parameter here in a field 2 chars wide in hex.*

*// %3d means insert next parameter here in a field 3 characters wide in decimal.*

*// %1c means insert next parameter here in a field 1 wide as a Unicode character.*

*// Everything else, including the [] and :s are just decorative text.*

out**.**printf**(** "result is: [%2x] : %3d : %1c\n"**,** c**,** c**,** c **);**

*// prints : result is:\_[41]\_:\_\_65\_:\_A*

*// In JDK 1.4- you would code that as:*

System**.**out**.**println**(** "result is: ["

**+** Stringtools**.**leftPad**(** Integer**.**toHexString( c )**,** 2 **)**

**+** "] : "

**+** Stringtools**.**leftPad**(** Integer**.**toString( c )**,** 3 **)**

**+** " : "

**+ (**char**)** c**);**

*// The JDK 1.5 code is considerably easier to write and proofread, but the 1.4 code is faster.*

*// Possible codes you might use include:*

*//*

*// %b boolean*

*// %h hashcode of the value. NOT hex!!*

*// %s String*

*// %c Unicode character*

*// %d decimal integer*

*// %o octal integer*

*// %x hex integer*

*// %e floating point scientific notation with exponent.*

*// %f floating point as a decimal number without exponent.*

*// %g general format %e or %f depending on size.*

*// %a precise hex floating number.*

*// %t date/time*

*// %% a literal %*

*// %n line separator, a platform-specific line separator, not necessarily \n.*

*// %0d left pad with 0*

*// %,d group with commas*

*// %-d left justify*

*// %+d explict sign, + or -.*

*// %% literal %*

*// see documentation on java.util.Formatter for a complete list and the full syntax.*

*// Use of G format:*

*// %[flags][min width of field].[number of significant digits of precision (not # of decimal places)]g*

System**.**out**.**printf**(** "value is %4.3g\n"**,** 1.4d **);** *// prints value is 1.40*

System**.**out**.**printf**(** "value is %4.3g\n"**,** 1.456d **);** *// prints value is 1.46*

System**.**out**.**printf**(** "value is %4.3g\n"**,** 145.0d **);** *// prints value is \_145*

System**.**out**.**printf**(** "value is %8.4g\n"**,** 1.5d **);** *// prints value is \_\_\_1.500*

System**.**out**.**printf**(** "value is %4.4g\n"**,** 1.5E-6 **);** *// prints value is 1.500e-06*

System**.**out**.**printf**(** "value is %,8.6g\n"**,** 12345.0d **);** *// prints value is 12,345.0*

import java.applet.\*;

/\*\* Illustrate use of the DecimalFormat class \*\*/

public class PrintfDemo extends Applet

{

/\*\* Illustrate several output formats with printf() \*\*/

public void init () {

double q = 1.0/3.0;

// Print the number with 3 decimal places.

System.out.printf ("1.0/3.0 = %5.3f %n", q);

// Increase the number of decimal places

System.out.printf ("1.0/3.0 = %7.5f %n", q);

// Pad with zeros.

q = 1.0/2.0;

System.out.printf ("1.0/2.0 = %09.3f %n", q);

// Scientific notation

q = 1000.0/3.0;

System.out.printf ("1000/3.0 = %7.2e %n", q);

// More scientific notation

q = 3.0/4567.0;

System.out.printf ("3.0/4567.0 = %7.2e %n", q);

// Negative infinity

q = -1.0/0.0;

System.out.printf ("-1.0/0.0 = %7.2e %n", q);

q = 0.0/0.0;

// NaN

System.out.printf ("0.0/0.0 = %5.2e %n", q);

// Multiple arguments

System.out.printf ("pi = %5.3f, e = %5.4f %n", Math.PI, Math.E);

double r = 1.1;

// User the argument index to put the argument values into

// different locations within th string.

System.out.printf ("C = 2 \* %1$5.5f \* %2$4.1f, "+

"A = %2$4.1f \* %2$4.1f \* %1$5.5f %n",

Math.PI, r);

}

/\*\* Paint message in Applet window. \*\*/

public void paint (java.awt.Graphics g) {

g.drawString ("Test printf()",20,20);

}

/\*\* Can also run this program as an application.

\* No windowing needed so just run the applets

\* init () function.

\*\*/

public static void main (String [] args) {

PrintfDemo obj = new PrintfDemo ();

obj.init ();

}

} // class PrintfDemo