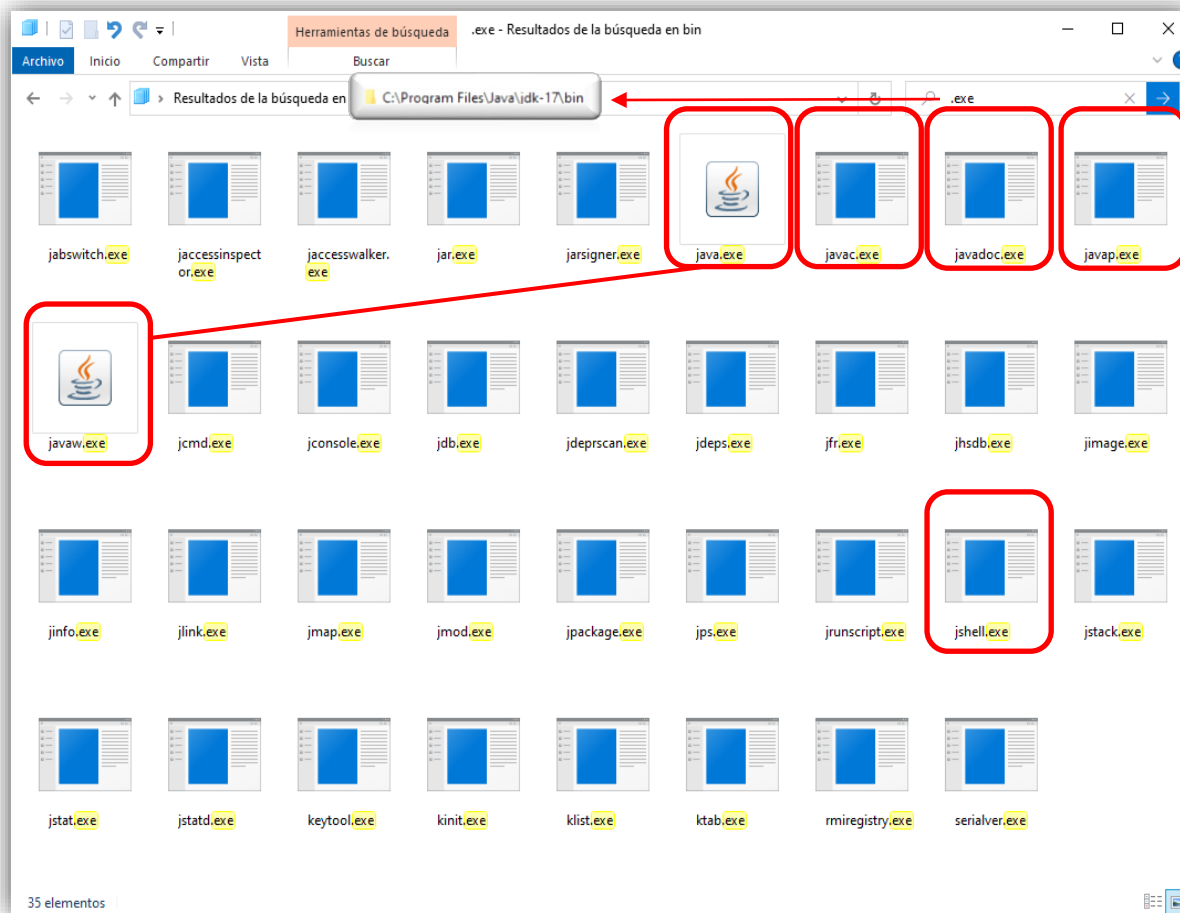


¿QUÉ TENDRÉ A MI DISPOSICIÓN PARA DESARROLLAR EN JAVA?

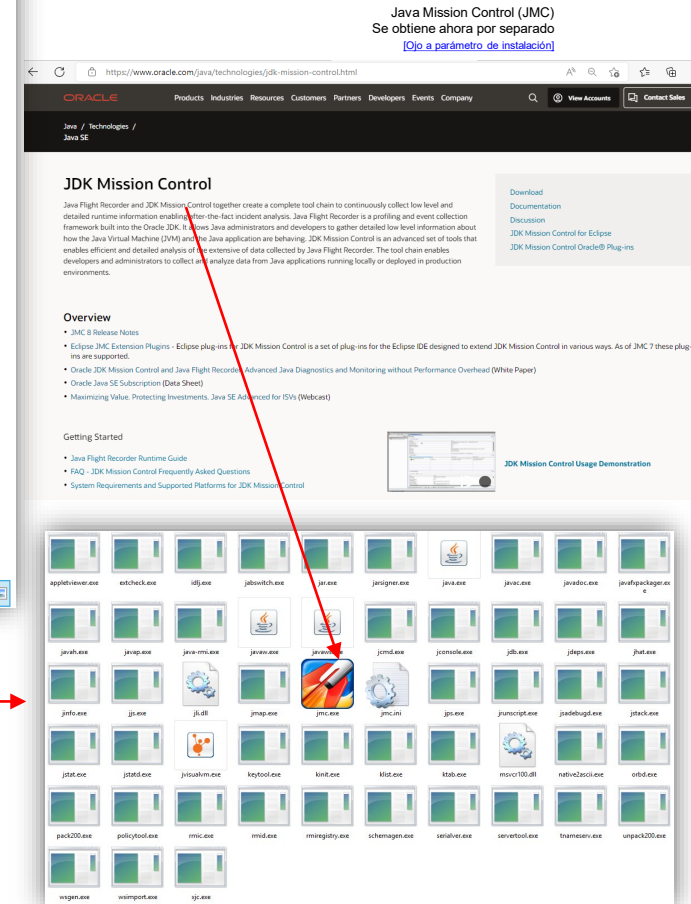
(COMPILACIÓN, EJECUCIÓN, ETC.)

Las herramientas de ejecución y desarrollo de Java

El directorio “/bin”



En una versión anterior... →
En distintas versiones van cambiando herramientas, pero en todo caso pueden obtenerse las que no se encuentran por defecto. Las imprescindibles las encontramos siempre.



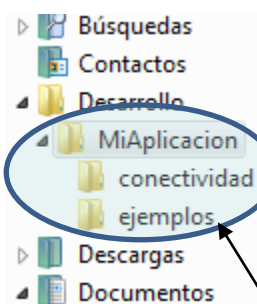
Compilación y ejecución

```
D:\>java
Usage: java [options] <mainclass> [args...]
       (to execute a class)
or java [options] -jar <jarfile> [args...]
       (to execute a jar file)
or java [options] -m <module>[/<mainclass>] [args...]
       (to execute the main class in a module)
or java [options] --module <module>[/<mainclass>] [args...]
       (to execute the main class in a module)
or java [options] <sourcefile> [args]
       (to execute a single source-file program)

Arguments following the main class, source file, -jar <jarfile>,
-m or --module <module>/<mainclass> are passed as the arguments to
main class.

where options include:

-cp <class search path of directories and zip/jar files>
-classpath <class search path of directories and zip/jar files>
--class-path <class search path of directories and zip/jar files>
      A ; separated list of directories, JAR archives,
      and ZIP archives to search for class files.
-p <module path>
--module-path <module path>...
      A ; separated list of directories, each directory
      is a directory of modules.
--upgrade-module-path <module path>
```



Un programa Java comienza por la ejecución de un fichero (una clase) que puede ir invocando la ejecución de otras dentro de uno o varios subárboles de directorios o ficheros JAR (cada subárbol o cada JAR es una “biblioteca de clases”)

HolaMundo.class
HolaMundo.jar
HolaMundo.java

```
D:\>java -version
java version "17" 2021-09-14 LTS
Java(TM) SE Runtime Environment (build 17+35-LTS-2724)
Java HotSpot(TM) 64-Bit Server VM (build 17+35-LTS-2724, mixed mode, sharing)
```

```
D:\>javac -version
javac 17
```

```
D:\>javac
Usage: javac <options> <source files>
where possible options include:
@<filename>          Read options and filenames from file
-Akey[=value]         Options to pass to annotation processors
--add-modules <module>(<module>)*
      Root modules to resolve in addition to the initial modules, or all modules
      on the module path if <module> is ALL-MODULE-PATH.
--boot-class-path <path>, -bootclasspath <path>
      Override location of bootstrap class files
--class-path <path>, -classpath <path>, -cp <path>
      Specify where to find user class files and annotation processors
-d <directory>        Specify where to place generated class files
-deprecation          Output source locations where deprecated APIs are used
--enable-preview      Enable preview language features. To be used in conjunction with either -source or --release.
-encoding <encoding> Specify character encoding used by source files
-endorseddirs <dirs>  Override location of endorsed standards path
-extdirs <dirs>       Override location of installed extensions
-g                  Generate all debugging info
-g:{lines,vars,source} Generate only some debugging info
-g:none            Generate no debugging info
-h <directory>       Specify where to place generated native header files
--help, -help, -?    Print this help message
--help-extra, -X     Print help on extra options
-implicit:{none,class}
      Specify whether or not to generate class files for implicitly referenced files
-J<flag>             Pass <flag> directly to the runtime system
--limit-modules <module>(<module>)*
      Limit the universe of observable modules
```

Compilar y ejecutar estando en “MiAplicacion”

```
>javac ejemplos/HolaMundo.java
>java ejemplos.HolaMundo
```

Compilar y ejecutar estando en otro directorio

```
>javac -cp c:/Desarrollo/MiAplicacion ejemplos/HolaMundo.java
>java -cp c:/Desarrollo/MiAplicacion ejemplos.HolaMundo
```

Ejecutar mediante un JAR

```
>java -cp c:/Desarrollo/MiAplicacion/ejemplos/HolaMundo.jar HolaMundo
>java -jar c:/Desarrollo/MiAplicacion/ejemplos/HolaMundo.jar
```

Comprobaremos todo esto... escribiendo el HolaMundo.java, compilando, ejecutando, desensamblando, decompilando (<http://www.javadecompilers.com/>)
Haremos lo mismo con Netbeans. (<http://java-decompiler.github.io/>)

**Importante: el uso de packages.

___ Lo contrario: ofuscación ___

Desensamblado de codebytes y decompilación

HolaMundo.java

```
public class HolaMundo {  
    public static void main(String[] args) {  
        System.out.println("Hola, mundo");  
    }  
}
```

HolaMundo.class

```
Ej%Q^@^@4^@'^
^@^F^@T ^@U^@V^@H^@W
^@^@Y^@G^@Z^@C^[^@^F^@init>^@^C)V^@^DCode^@^OLineNumberTable^@
^@RLocalVariableTable^@^Dthis^@^ULAppTest3D/HolaMundo;^@^Dmain^@^U(
[Ljava/lang/String;)U^@^Dargs^@^S[java/lang/String;^@^D
SourceFile^@^NHolaMundo.java^@L^@G^@H^@G^@V^@L^@J]^@^^^@KHola, mundo^@
^@_L^@^@!^@^SappTest3D/HolaMundo^@^Pjava/lang/Object^@^Pjava/lang/
System^@^Cout^@^U^@L^@java/io/PrintStream;^@^S^@java/io/PrintStream^@^G^@pr
intln^@^U(L^@java/lang/String;)U^@!^@^@F^@^@^@B^@^@^@G^@H^@^@A^@
^@^@^@/^@^@^@^@E^@*^@^@±^@^@^@B^@
^@^@^@F^@^@^@L^@K^@^@^@L^@^@^@E^@^@L^@H^@^@^@ ^@N^@O^@
^@^@^@7^@B^@^@^@ z^@B^@R^@C^@^@D±^@^@^@B^@
^@^@^@
^@B^@^@q^@^@r^@k^@^@l^@^@^@ ^@P^@q^@^@r^@^@
B^@S
```

```
C:>javap -c HolaMundo
```

Compiled from "HolaMundo.java"

```
public class HolaMundo extends java.lang.Object{
    public HolaMundo();
```

Code :

```
0:  aload 0
```

```
1: invokespecial #1; //Method java/lang/Object."<init>":()V
```

```
4: return
```

```
public static void main(java.lang.String[]);
```

Code:

```
0:  getstatic      #2; //Field java/lang/System.out:Ljava/io/PrintStream;
```

```
3: ldc      #3; //String Hola, mundo
```

```
5:  invokevirtual   #4; //Method java/io/PrintStream.println: (Ljava/lang/String;)V
```

```
8:  return
```

}

	Magic number		Minor version		Mayor version		
00000000:	cafe	babe	0000	0034	0022	0a00	0600 1409
00000010:	0015	0016	0000	170a	0108	0019	0700 1a07
00000020:	001b	0100	063c	696e	6974	3e01	0003 2829
00000030:	5601	0004	436f	6465	0100	0f4c	696e 654e
00000040:	756d	6265	7254	6162	6c65	0100	124c 6f63
00000050:	616c	5661	7269	6162	6665	5461	626c 6501
00000060:	0004	7468	6973	0100	154c	6170	7054 6573
00000070:	7433	442f	486f	6c61	0475	6e64	6f3b 0100
00000080:	046d	6169	6e01	0016	285b	4c6a	6176 612f
00000090:	6c61	6e67	2f53	7472	696e	673b	2956 0100
000000a0:	0461	7267	7301	0013	5b4c	6a6b	7661 2f6c
000000b0:	616e	672f	5374	7269	6e67	3b01	000a 536f
000000c0:	7572	6365	4669	6c65	0100	0e48	6f6c 614d
000000d0:	756e	646f	2e6a	6176	610c	0007	0008 0700
000000e0:	1c0c	001d	001e	0100	0b48	6f6c	612c 286d
000000f0:	756e	646f	0007	1f0c	0020	0021	0100 1361
00000100:	7070	5465	7374	3344	2f48	6f6c	614d 756e
00000110:	646f	0100	106a	6176	612f	6c61	6e67 2f4f
00000120:	626a	6563	7401	0010	6a61	7661	2f6c 616e
00000130:	672f	5379	7374	656d	0100	036f	7574 0100
00000140:	154c	6a61	7661	2f69	6f2f	5072	696e 7453
00000150:	7472	6561	6d3b	0100	136a	6176	612f 696f
00000160:	2f50	7269	6e74	5374	7268	616d	0100 0770
00000170:	7269	6e74	6c6e	0100	1255	4c6a	6176 612f
00000180:	6c61	6e67	2f53	7472	696e	673b	2956 0021
00000190:	0005	0006	0000	0000	0002	0001	0007 0008
000001a0:	0001	0009	0000	002f	0001	0001	0000 0005
000001b0:	2ab7	0001	1100	0000	0200	0a00	0000 0600
000001c0:	0100	0000	0c00	0b00	0000	0c00	0100 0000
000001d0:	0500	0c00	0d00	0000	0900	0e00	0f00 0100
000001e0:	0900	0000	3f00	0200	0100	0000	09b2 0000
000001f0:	1203	b600	04b1	0000	0002	000a	0000 000a
00000200:	0002	0000	0011	0008	0012	000b	0000 000c
00000210:	0001	0000	0009	0010	0011	0000	0001 0012
00000220:	0000	0002	0013	0a0a	0a0a	0a0a	

Java SE	23	= 67	(0x43)
Java SE	22	= 66	(0x42)
Java SE	21	= 65	(0x41)
Java SE	20	= 64	(0x40)
Java SE	19	= 63	(0x3F)
Java SE	18	= 62	(0x3E)
Java SE	17	= 61	(0x3D)
Java SE	16	= 60	(0x3C)
Java SE	15	= 59	(0x3B)
Java SE	14	= 58	(0x3A)
Java SE	13	= 57	(0x39)
Java SE	12	= 56	(0x38)
Java SE	11	= 55	(0x37)
Java SE	10	= 54	(0x36)
Java SE	9	= 53	(0x35)
Java SE	8	= 52	(0x34)
Java SE	7	= 51	(0x33)
Java SE	6	= 50	(0x32)
J2SE 5.0 (1.5)	= 49	(0x31)	
J2SE 1.5	= 48	(0x30)	
J2SE 1.3	= 47	(0x2F)	
J2SE 1.2	= 46	(0x2E)	
JDK 1.1	= 45	(0x2D)	
JDK 1.0	= 44	(0x2C)	

Sobre el “Magic Number”

We used to go to lunch at a place called St Michael's Alley. According to local legend, in the deep dark past, the Grateful Dead used to perform there before they made it big. It was a pretty funky place that was definitely a Grateful Dead Kinda Place. When Jerry died, they even put up a little Buddhist-esque shrine. When we used to go there, we referred to the place as Cafe Dead. Somewhere along the line, it was noticed that this was a HEX number. I was re-vamping some file format code and needed a couple of magic numbers: one for the persistent object file, and one for classes. I used CAFEDEAD for the object file format, and in grepping for 4 character hex words that fit after "CAFE" (it seemed to be a good theme) I hit on BABE and decided to use it. At that time, it didn't seem terribly important or destined to go anywhere but the trash can of history. So CAFEABABE became the class file format, and CAFEDEAD was the persistent object format. But the persistent object facility went away, and along with it went the use of CAFEDEAD – it was eventually replaced by RMI.

James Gosling

