Introduction to Java

autor：Martin Ngobye.

Java is designed to meet the challenges of application development in the context of heterogeneous, network-wide distributed environments. Paramount among these challenges is secure delivery of applications that consume the minimum of system resources, can run on any hardware and software platform, and can be extended dynamically.

Java originated as part of a research project to develop advanced software for a wide variety of network devices and embedded systems. The goal was to develop a small, reliable, portable, distributed, real-time operating platform. When the project started, C++ was the language of choice. But over time the difficulties encountered with C++ grew to the point where the problems could best be addressed by creating an entirely new language platform. Design and architecture decisions drew from a variety of languages such as Eiffel, SmallTalk, Objective C, and Cedar/Mesa. The result is a language platform that has proven ideal for developing secure, distributed, network based end-user applications in environments ranging from network-embedded devices to the World-Wide Web and the desktop.

The design requirements of Java are driven by the nature of the computing environments in which software must be deployed.

The massive growth of the Internet and the World-Wide Web leads us to a completely new way of looking at development and distribution of software. To live in the world of electronic commerce and distribution, Java must enable the development of secure, high performance, and highly robust applications on multiple platforms in heterogeneous, distributed networks.

Operating on multiple platforms in heterogeneous networks invalidates the traditional schemes of binary distribution, release, upgrade, patch, and so on. To survive in this jungle, Java must be architecture neutral, portable, and dynamically adaptable.

The Java system that emerged to meet these needs is simple, so it can be easily programmed by most developers; familiar, so that current developers can easily learn Java; object oriented, to take advantage of modern software development methodologies and to fit into distributed client-server applications; multithreaded, for high performance in applications that need to perform multiple concurrent activities, such as multimedia; and interpreted, for maximum portability and dynamic

capabilities.

Together, the above requirements comprise quite a collection of buzzwords, so let's examine some of them and their respective benefits before going on.

What's completely new is the manner in which Java and its run-time system have combined them to produce a flexible and powerful programming system..

Developing your applications using Java results in software that is portable across multiple machine architectures, operating systems, and graphical user interfaces, secure, and high performance, With Java, your job as a software developer is much easier—you focus your full attention on the end goal of shipping innovative products on time, based on the solid foundation of Java. The better way to develop software is here, now, brought to you by the Java language platform.

Very dynamic languages like Lisp, TCL, and SmallTalk are often used for prototyping. One of the reasons for their success at this is that they are very robust—you don't have to worry about freeing or corrupting memory.

Similarly, programmers can be relatively fearless about dealing with memory when programming in Java, The garbage collection system makes the programmer's job vastly easier; with the burden of memory management taken off the programmer's shoulders, storage allocation errors go away. Another reason commonly given that languages like Lisp, TCL, and SmallTalk are good for prototyping is that they don't require you to pin down decisions early on—these languages are semantically rich.

Java has exactly the opposite property: it forces you to make explicit choices. Along with these choices come a lot of assistance—you can write method invocations and, if you get something wrong, you get told about it at compile time. You don't have to worry about method invocation error.

The Java beginner must grasp 30 basic concepts

Basic concept

1.In OOP the only relations is what the object' s interface, such as the computer seller he leaves alone internal structure of electrical source, but he is only concerned about that whether the power will be supplied to you, also so long as know can or not but is not how and why.All procedures are make up of certain attribute and the behavior object, the different object visit invokes through the function invocation, between the object all intercommunion are invoke through the method invocation, By object data encapsulation, enhances reuse rate very much.

2.In OOP the most important thought is class, the class is the template ,is a

blueprint, construct an object from a class, namely construct an instance of the class.

3. Encapsulation: is that combines the data and the behavior in a package) and hides the data the realization process to the object user, in an object data is called its instance field.

4.Through expands a class to obtain a new class is called inheritance, but all classes are constructed by the object super root class of expansion, super root class of as follows can make the introduction.

5.Object 3 principal characteristics

Behavior--- explained this object can make what.

Tate--- when the object exerts the method object reflection.

Dentity--- and other similar behavior objects discrimination symbols.

Each object has only indentity and among three characteristics they affect mutually.

6. Relations among classes:

Use-a:Dependent relation

Has-a:Polymerization relation

Is-a: inheritor relation -- example:A class has inherited B class, this time A class not only has B class of method, but also has its own method(Individuality exists in general character)

7.Structure object use structure:Structure proposing, the structure is one special method, the structure object and to its initialization.

Example:A Data class of structure calls Data

New Data () --- structure a new object, also initialize current time.

Data happyday=new Data () --- an object evaluates an variable happyday, thus enables this object to be used many times, here be stated the cause variable and the object variable are different.

New returns the value is a quotation.

Constructor characteristic:The constructor may have 0, one or many parameters

The constructor and the class have the same name

A class may have many constructor. The constructor has not returned value

The constructor always be together used with the new operator

8. Over loading: When many methods have the same name when includes the different parameter, then has the over loading Which method does the compiler

have to choose invokes.

9.Package : Java allow one or many classes to become together as group, is called package, to organizing duty easily, the standard Java storehouse divides into many packages Java.Lang java.Util java, net and so on, the package is layered and all java packages are in java and in a javax package.

10. Extendable thought: permit to construct new class on existing classes , when you extend the class which already existed, then you reuse this class of method and the field, at the same time you might add the new method and the field in the new class.

11.Expandable class:The expandable class fully manifested is-a to extend the relations The form is:Class (subclass) extends (base class).

12. Multi-modality: In java, the object variable is multi-modality But in java does not support multiple extend.

13.Dynamic combine: the mechanism of invoking object method mechanism.

1) compiler examines object statement type and method name.

2) the compiler examines that method invokes parameter type.

3) static combine: If the method type is priavte static the final ,compiler can accurately know which method should invoke.

4) when the procedure runs and uses dynamic combine to invoke a method, the method edition which then hypothesized machine must invoke x the object actual type which aims at to match.

5) dynamic combine: is a very important characteristic, it can cause the procedure to change again may expand but does not need to translate has saved the code.

14.Final class:In order to prevent other people derive the new class from yours class, this class is cannot expanded.

15.The dynamic invocation spend longer time than the static invocation expenditure.

16.Abstract class:Stipulated or many abstract methods class of itself must define is abstract.

Example: Public abstract string getDescripition

17.In Java each class is be extended by object class.

18. equal and toString method in object class .

Equal uses in testing an object is whether equal with another object.

ToString returns to represent this object the string of character, each class can nearly over loading this method, in order to returns to the current condition the correct expression.

(The toString method is a very important method)

19.General programming:Any class of type all values all may replace with a object class of variable.

20.The array tabulates: The ArrayList dynamic array tabulates, is a class of storehouse, defines in java.In uitl package, but automatic control array size.

21.in class and class of object ,getclass method returns to the class type an example, when the procedure start contains can increase in the main method class, hypothesized confidential increase all classes which he needs, each increase class all must increase the class which it needs.

22.The class: class might dynamic operate the java code for the compilation the procedure to provide the formidable function reflection, this function was JavaBeans is specially useful, the use reflected Java to be able to support the VB programmer to be familiar with the use the tool.

procedure of analysis class ability is called the reflector, in Java to provide this function the package to call Java.Lang.The reflect reflection mechanism is extremely formidable.

1) when run analysis class ability.

2) when run searches observes a class of object.

3) realizes the general array operation code.

4) provides the method object.

But this mechanism mainly aims at the tool but not the application and the procedure.

In the reflection mechanism most important part is that permits class that you inspect structure. With to API includes:

Java.Lang.Reflect.Field returns to the field.

Java.Reflect.Method returns to the method.

Java.Lang.Reflect.Constructor returns to the parameter.

Method pointer: Java does not have the method pointer, makes a method address another method, may invoke it in behind, but the interface is the better solution.

23. interface: should showing class could do what but not to assign how to do, a class may realize one or many interfaces.

24.The interface is not a class, but is to conforms to a interface request class of set of standard.

If realizes a interface to need 2 steps:

1) the statement class needs to realize assigns the interface.

2) provides in the interface all methods definition.

Stated a class realizes a interface to need to use the implements key words

Class actionB implements Comparable its actionb needs to provide the CompareTo method, the interface is not the class, cannot use a new example interface.

25.A class only then a super class, but a class can realize many interfaces. In a Java important interface: Cloneable

26.The interface and call-back :to programs a commonly used pattern is call-back,in the pattern, you may refer when this class of pattern settled specific time occurs returns to adjusts on the object the method.

Example:ActionListener interface monitor.

Similar API includes:

Java.Swing.JOptionPane

Java.Swing.Timer

Java.Awt.Tookit

27.Object clone:The clone method is a object protection method, this meant your code cannot simple invoke it.

28.Inner class an inner class definition is the definition in another class.

The reason is:

1) an inner class object can visit founds its object realization, including private data.

2) about other classes in the same package in that, inner class can be hided.

3) the anonymous inner class may the very convenient definition accent.

In 4) uses the category to be possible the extremely convenient compilation event driver.

29.Agent class (proxy):

1) appointing all codes that interfaces request

2) all methods (toString equals) that object class define

30.Data type:Java is kind of emphasizing type language, each variable all

must be declared its types at first, in java altogether has 8 basic types . four kinds are the long, two kinds are the float, one is the char, being used in the Unicode code char, Boolean.

1. java is simple

Java and C + + are very similar, but much simpler. All the high-level programming language features, is not absolutely necessary have been deleted. For example, Java does not overload operator, the title of the document, pre-processing, computing pointer, the structure of the joint, multi-dimensional array, templates and implicit type conversion. If you know a little C, C + + or Pascal, you will soon master Java. Here is a simple procedure JavaHelloWorld:

```
publicclassHelloInternet (
publicstaticvoidmain (Stringargv []) (
System. out. println ( "HelloIn-ternet!")
)
)
```

2. Java is object-oriented

Java is an object-oriented programming language. In addition to the simple types, such as digital and Boolean operators in addition, Java is an object of most. As with any object-oriented languages, Java code also organized by category. Each category provides a definition of the object behavior. Another type of succession can be a kind of behavior. In the category of the root level, often the target category. Java support for the single type of inherited hierarchy. This means that each category can only inherit one other category. Some of the language to allow multiple inheritance, but it may cause confusion and unnecessarily complicated language. For example, imagine that an object would inherit two completely different category. Java also supports the kind of summary of the interface. This allows programmers to define the interface methods, and do not have to rush immediately to determine the methods to achieve. A type of interface can be a number of purposes in order to truly multi-inheritance of a number of advantages. The implementation of an object can be any number of interfaces. IDL interface and Java interface very similar. Very easy to set up IDLJ compiler. In other words, Java can be used to create a CORBA object system distributed object system. In the view of many computer systems use IDL interface and CORBA objects, such compatibility is important.

3. Java is a type of static

In a Java program, it is essential to the definition used by the target (number of characters, such as an array) type. This will help programmers quickly found because the procedure when the compiler can detect the type of error. However, Java System objects are also dynamic types. A requirement for the type of dynamic is often possible, so programmers can write the procedures for different types of objects to do different things.

4. Is a Java-based compiler

When running Java programs, which were first compiled into byte code. Byte code is very similar to the machine instructions, so Java program is very efficient. However, the byte code does not specifically for a particular machine, so no need to recompile Java program can be in many different computer implementation. Java source code files were compiled into a category, which is equivalent to process byte code performance. In a Java class file, and an example for all of the variables are in the light of, and for the first time in the implementation of the code be resolved. This makes the code more common and more easily subject to revision, but still high.

5. Java is architecture neutral

Java language is the same for each computer. For example, simple types are the same: 32-bit integer always, always 64-bit long integers. It is strange, such as C and C + + programming language, and so fashionable it is not the case. As a result of these languages so the definition of freedom, each of the compiler and development environment will be different, so that this process nuisance become a transplant. Java programs can easily gain access to transplants, and there is no need to re-compile.

6. Java is a sound

Java program can not be caused by the collapse of the computer. Java careful testing of the system memory of each visit, make sure it is legitimate and will not cause any problems. However, even if the Java program may also be wrong. If there is some kind of unexpected things, the process will not collapse, and to abandon the exception. Procedures for such an exception would be found to address them. Traditional computer programs can access the full memory. May (unconsciously) to amend any of the value of memory, which will cause problems. Java program can only access memory to allow them access to those parts of the Java program can not modify it does not seek to change the value.

7. Java is a compact

As the Java is designed to run on a small computer, as a programming language for the system is relatively small. It effectively in more than 4MB of RAM to run on PC machine. Java translator occupied by only a few hundred KB. This translator for the Java platform independence and portability is reliable. Due to Java is very small, it is a very small computer memory, such as the Java-based PC, as well as television sets, ovens, telephone and home computer, and so on, it is ideal.

8. Java is a multi-threaded

Java program can run more than one thread. For example, it can be a thread in a time-consuming to complete the calculation, and other users to interact with the threads of dialogue. Therefore, users do not have to stop working, waiting for the Java computing time-consuming process. In the multi-threaded programming environment, it is often difficult because many things may occur at the same time. However, Java provides easy-to-use features simultaneously, so that the programming easier.