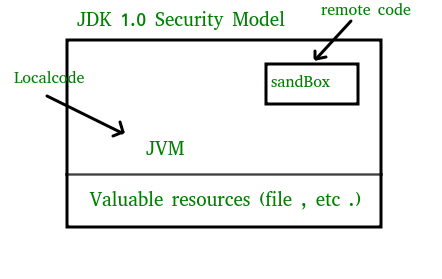
**Sandbox Security Model**

In computer security, a sandbox is a security mechanism for separating running programs, usually in order to minimize system failures or software vulnerabilities from spreading. In general, a sandbox is an isolated computing environment in which a program or file can be executed without affecting the application in which it runs. Sandboxes are used by software developers to test new programming code.

* A sandbox typically provides a tightly controlled set of resources for guest programs to run in, such as limited space on disk and memory.
* In a Java programming language, the sandbox is the program area and it has some set of rules that programmers need to follow when creating Java code (like an applet) that is sent as part of a page.
* Since a Java applet is sent automatically as part of the page and can be executed as soon as it arrives, the applet can easily do harm, either accidentally or intentionally
* The sandbox restrictions provide strict limitations on which system resources the applet can request or access.The programmer must write code that “plays” only within the sandbox, such as children are allowed to play within the confined limits of a place. The sandbox can be conceived as a small area within your computer where an applet’s code can play freely – but it’s not allowed to play anywhere else.

**Security in Java**

* The original security model provided by the Java platform is known as the sandbox model, which existed in order to provide a very restricted environment in which to run untrusted code obtained from the open network.
* The essence of the sandbox model is that local code is trusted to have full access to vital system resources (such as the file system) while downloaded remote code (an applet) is not trusted and can access only the limited resources provided inside the sandbox.



* The sandbox model was deployed through the Java Development Kit (JDK), and was generally adopted by applications built with JDK 1.0, including Java-enabled web browsers.
* Overall security is provided through a number of mechanisms.The language is designed to be type-safe and easy to use i.e the hope is that the burden on the programmer is such that the likelihood of making mistakes is less compare to using other programming languages such as C or C++. Language features such as automatic memory management, garbage collection, and range checking on strings and arrays are examples of how the language helps the programmer to write safe code.
* Second, compilers and a bytecode verifier ensure that only legitimate Java bytecodes are executed. The bytecode verifier, together with the Java Virtual Machine, guarantees language safety at run time.
* A Classloader defines a local name space, which can be used to ensure that an untrusted applet cannot interfere with the running of other programs.
* Finally, access to crucial system resources is mediated by the Java Virtual Machine and is checked in advance by a SecurityManager class that restricts the actions of a piece of untrusted code to the bare minimum.**(SandBoxing)**

**Usage**

Sandboxing is frequently used to test unverified programs that may contain a virus or other malicious code, without allowing the software to harm the host device.

**Implementations**

* Online judge systems to test programs in programming contests.
* New-generation pastebins allowing users to execute pasted code snippets on the pastebin’s server.
* HTML5 has a “sandbox” attribute for use with iframes.
* Secure Computing Mode(seccomp) is a sandbox built in the Linux kernel. When activated, seccomp only allows the write(), read(), exit(), and sigreturn() system calls.