# [Why character array is better than String for Storing password in Java](http://javarevisited.blogspot.in/2012/03/why-character-array-is-better-than.html" \o "Why character array is better than String for Storing password in Java)

**Why character array is better than String for storing password in Java** was recent question asked to one of my friend in a java interview. he was interviewing for a Technical lead position and has over 6 years of experience.Both [Character array and String](http://javarevisited.blogspot.com/2012/02/how-to-convert-char-to-string-in-java.html) can be used to store text data but choosing one over other is difficult question if you haven't faced the situation already. But as my friend said any question related to String must have a clue on special property of Strings like immutability and he used that to convince interviewer. here we will explore some reasons on why should you used char[] for storing password than String.  
  
[character array over string storing password java](http://javarevisited.blogspot.com/2011/08/convert-string-to-integer-to-string.html)This article is in continuation of my earlier interview question post on String e.g. [Why String is immutable in Java](http://javarevisited.blogspot.com/2010/10/why-string-is-immutable-in-java.html) or [How Substring can cause memory leak in Java](http://javarevisited.blogspot.com/2011/10/how-substring-in-java-works.html), if you haven't read those you may find them interesting.Here are few reasons which makes sense to believe that character array is better choice for storing password in Java than String:

1) Since **Strings are immutable in Java** if you store password as plain text it will be available in memory until Garbage collector clears it and since String are used in String pool for reusability there is pretty high chance that it will be remain in memory for long duration, which pose a security threat. Since any one who has access to memory dump can find the password in clear text and that's another reason you should always used an encrypted password than plain text. Since Strings are immutable there is no way contents of Strings can be changed because [any change will produce new String](http://javarevisited.blogspot.com/2011/07/string-vs-stringbuffer-vs-stringbuilder.html), while if you char[] you can still set all his element as blank or zero. So **Storing password in character array clearly mitigates security risk of stealing password**.

2) **Java itself recommends**using getPassword() method of JPasswordField which returns a char[] and deprecated getText() method which returns password in clear text stating security reason. Its good to follow advice from Java team and adhering to standard rather than going against it.

3) With String there is always a risk of printing plain text in [log file or console](http://javarevisited.blogspot.com/2011/05/top-10-tips-on-logging-in-java.html) but if use [Array](http://javarevisited.blogspot.com/2012/01/anonymous-array-example-java-create.html) you won't print contents of array instead its memory location get printed. though not a real reason but still make sense.

[**String**](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html) strPassword="Unknown";  
**char**[] charPassword= **new** **char**[]{'U','n','k','w','o','n'};  
**System**.out.println("String password: " + strPassword);  
**System**.out.println("Character password: " + charPassword);  
  
**String** password: Unknown  
**Character** password: [C@110b053

That's all on *why character array is better choice than String for storing passwords in Java*.  Though using char[] is not just enough you need to erase content to be more secure. I also suggest working with hash'd or [encrypted password](http://javarevisited.blogspot.com/2012/02/how-to-encode-decode-string-in-java.html) instead of plain text and clearing it from memory as soon as authentication is completed.