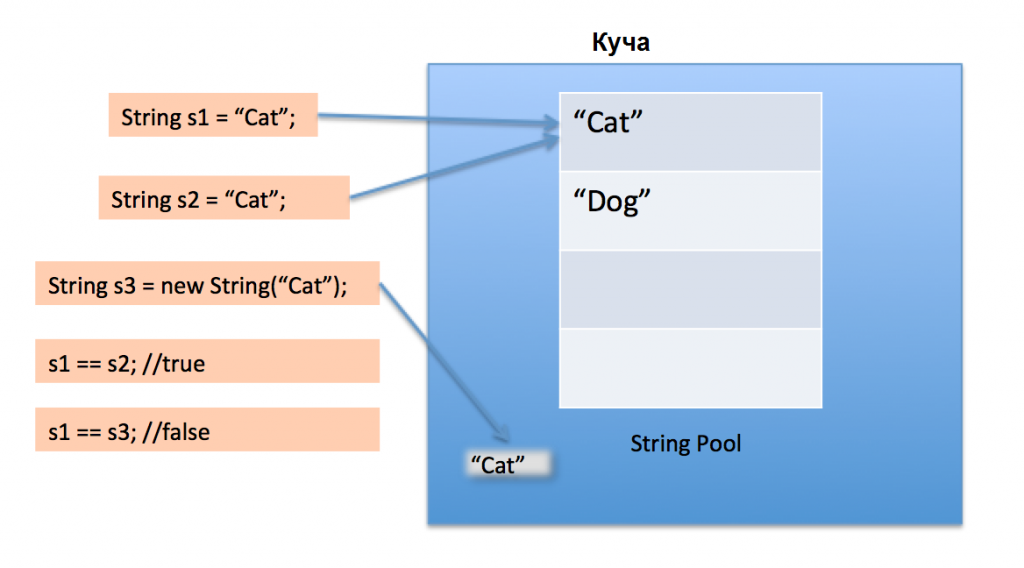
<https://javadevblog.com/chto-takoe-pul-strok-v-java.html>

Пул строк (****String Pool****) — это множество строк в кучи (**[Java Heap Memory](https://javadevblog.com/chto-takoe-heap-i-stack-pamyat-v-java.html" \t "https://javadevblog.com/_blank)**). Мы знаем, что String — особый класс в java, с помощью которого мы можем создавать строковые объекты.

На диаграмме ниже мы видим как именно строковый пул расположен в памяти Java Heap. И как разные способы создания строк влияют на расположение их в памяти.

[](https://javadevblog.com/wp-content/uploads/2015/07/String-Pool-Java1.png)

Сам строковый пул возможен только потому, что [строки в Java неизменные](https://javadevblog.com/pochemu-stroki-v-java-neizmennye.html). Также пул строк позволяет сохранить память в Java Runtime, хотя это и требует больше времени на создание самой строки.

Когда мы используем двойные кавычки, чтобы создать новую строку, то первым делом идет поиск строки с таким же значением в пуле строк. Если java такую строку нашла, то возвращает ссылку, в противном случае создается новая строка в пуле, а затем возвращается ссылка.

Однако использование оператора new заставляет класс String создать новый объект String. После этого можем использовать метод intern(), чтобы поместить этот объект в пул строк или обратиться к другому объекту из пула строк, который имеет такое же значение.

Метод intern() - читать  
<https://habr.com/ru/post/79913/>

Интернированные строки не хранятся вечно. Строки, на которых нет ссылок, также удаляются сборщиком мусора.  
В большинстве случаев вы не получите существенного прироста производительности от использования intern() — если сравнение строк не является основной (или очень частой) операцией вашего приложения и сравниваемые строки разные по длине.