Chapter 1: Text Blocks, Locales, Numbers, and Math

1. Creating a multiline SQL, JSON, and HTML string

Write a program that declares a multiline string (for instance, SQL, JSON, and HTML strings).

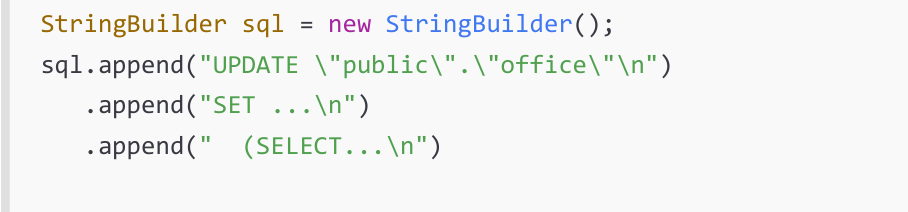
Let’s consider the following SQL multiline string:

Before JDK 8

Probably the most common approach relies on straightforward concatenation via the well-known “+" operator. This way, we obtain a multiline string representation, as follows:



The compiler should be (and usually is) smart enough to internally transform the “+" operations into a StringBuilder/StringBuffer instance and use the append() method to build the final string. However, we can use StringBuilder (not thread-safe) or StringBuffer (thread-safe) directly, as in the following example:



All the previous examples have a bunch of shortcomings in common. The most important of these is that none of these examples represents a truly multiline string literal, and the degree of readability is seriously affected by the escaping characters and extra quotes needed for each line demarcation. Fortunately, starting with JDK 13 (as a future preview) and ending with JDK 15 (as a final feature), the new text blocks have become the standard for representing multiline string literals. Let’s see how.

Introducing text blocks (JDK 13/15)

JDK 13 (JEP 355) offers a preview feature that aims to add support for multiline string literals. Over

two versions, in JDK 15 (JEP 378), the text block feature has become final and permanent for use. But

that’s enough history; let’s quickly see how text blocks shape our multiline SQL string:

String sql="""

UPDATE "public"."office"

SET ("address\_first", "address\_second", "phone") =

(SELECT "public"."employee"."first\_name",

"public"."employee"."last\_name", ?

FROM "public"."employee"

WHERE "public"."employee"."job\_title" = ?""";

The text block is concise, easy to update, and easy to understand

Here is another example that embeds a piece of JSON information:

String json = """

{

"widget": {

"debug": "on",

"window": {

"title": "Sample Widget 1",

"name": "back\_window"

},

"image": {

"src": "images\\sw.png"

},

"text": {

"data": "Click Me",

"size": 39

}

}

}""";

How about representing a piece of HTML as a text block? Sure, here it is:

String html = """

<table>

<tr>

<thcolspan="2">Name</th>

<th>Age</th>

</tr>

<tr>

<td>John</td>

<td>Smith</td>

<td>22</td>

</tr>

<table>""";

Hooking text blocks syntax

The syntax of text blocks is quite simple. No bells and whistles, no complicated things – there are just

Two aspects to keep in mind:

* A text block must start with """ (that is, three double quotation marks) and a newline. We refer to this construction as the opening delimiter.
* A text block must end with """ (that is, three double quotation marks). The """ can be on its own line (as a new line) or at the end of the last line of text (as in our example). We refer to this construction as the closing delimiter.

**In this context, the following examples are syntactically correct:**

String tb = """

I'm a text block""";

String tb = """

I'm a text block

""";

String tb = """

I'm a text block""";

String tb = """

I'm a text block

""";

String tb = """

I'm a text block

""";

**On the other hand, the following examples are incorrect and lead to compiler errors:**

String tb = """I'm a text block""";

String tb = "I'm a text block""";

String tb = """I'm a text block";

String tb = ""I'm a text block""";

String tb = """I'm a text block"";

String tb = ""I'm a text block

""";