Placeholders

1. Placeholders

Before describing what JaxMeJS placeholders are, we recall the use of placeholders in JDBC: By preparing a statement like

```
INSERT INTO someTable VALUES (?, ?, ?)
```

you create the statement at one point. At some other point (typically immediately after creating the statement, but not necessarily so) you add parameters to the statement. For example, you tell the JDBC driver: "Replace the first question mark (the first placeholder) with the string 'foo'". Likewise, you would say: "Replace the second question mark with the integer 5".

A JaxMeJS placeholder is quite comparable to a JDBC placeholder. The main idea is that you create one part of a method at some place and another part elsewhere. The method creating the other part must know where to add code. It finds the right place by using the placeholder, which the first part has set.

2. Creating a placeholder

A placeholder is created by invoking the method newPlaceHolder(String,boolean). For example:

```
  JavaMethod jm;
  PlaceHolder p = jm.newPaceHolder("MyPlaceholder", true);
```

The method takes two arguments:

pName
Any placeholder has got a name. The name is the identifier by which you find the placeholder. (See the next section for details.)

pAutoRemovable
The boolean argument indicates whether you expect to use the placeholder in either case. If so, you set the "autoRemovable" argument to false. In that case, you *must* remove the placeholder before serializing the method. An IllegalStateException is thrown otherwise. However, if you set the

"pAutoRemovable" argument to true, then an existing placeholder is simply ignored while serializing the method.

3. Using the placeholder

To insert code after the placeholder, simply find it. For example:

```
JavaMethod jm;
PlaceHolder p = jm.getPlaceHolder("MyPlaceholder");
jm.addLine("// This line follows immediately after the placeholder.");
```

Do not forget to remove the placeholder, if the "pAutoRemovable" flag was set to false while creating it:

```
p.remove();
```