**Переменные в Java**

**Переменная в Java (как и в других языках)** это основной элемент хранения информации. Переменная характеризуется комбинацией типа, идентификатора и области действия.

Как часть текста программы, переменная предоставляет программисту возможность обращаться к тому или иному участку памяти по идентификатору понятному для человека.

**Java может создавать переменные как в куче, так и на стеке. В стеке хранятся только примитивные типы. Все объекты создаются в куче (для этого используется оператор new) и передаются по ссылке.**

**Примитивные типы в Java**

**В Java представлены такие примитивные типы:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Type** | **Size, bit** | **Min** | **Max** | **Wrapper** | **Usage** |
| boolean | 1 | — | — | Boolean | Предназначен для хранения логических значений. Может быть только true или false. |
| char | 16 | Unicode 0 | Unicode 216- 1 | Character | Представляет собой один 16-битный Unicode символ |
| byte | 8 | -128 | +127 | Byte | Полезен при работе с потоком данных, который поступает из сети или файла |
| short | 16 | -215 | +215-1 | Short | Двухбайтовые целые числа |
| int | 32 | -231 | +231-1 | Integer | Наиболее часто используется при работе с целыми числами. |
| long | 64 | -263 | +263-1 | Long | Удобен для работы с большими целыми числами |
| float | 32 | 1,4\*10-45 | 3,4\*1038 | Float | Удобен для использования, когда не требуется особой точности в дробной части числа. |
| double | 64 | 4,9\*10-324 | 1,8\*10308 | Double | Дробные числа высокой точности |
| void | — | — | — | Void |  |

**Использование переменной определяется ее типом. Java осуществляет строгую типизацию, поэтому типы должны использоваться по назначению, несмотря на то что преобразования между типами возможны (кроме boolean).**

**Все численные типы в Java являются знаковыми.**

**При переполнении типа в сторону максимальных значений переменной присваивается минимальное значение диапазона. При переполнении типа в сторону минимальных значений переменной присваивается максимальное значение диапазона. Таким образом значения переменных «зацикливаются» при переполнении. Например: byte** testVariable = -128; --testVariable; будет 127.