**Жесткая типизация**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

[Уровень 10](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=10), Лекция 1

— Привет, Амиго!

— Привет, Элли!

— Сегодня у меня хорошее настроение, поэтому расскажу тебе кое-что интересное. И начну я с типизации примитивных типов в Java.

— В Java у каждого объекта и у каждой переменной есть свой жёстко заданный неизменяемый тип. Тип переменной определяется ещё в процессе компиляции программы, тип объекта – при его создании. Тип нового созданного объекта и/или переменной остаются неизменными все их время жизни. Пример:

|  |  |
| --- | --- |
| **Код на Java** | **Описание** |
| int a = 11;  int b = 5;  int c = a / b; // c == 2 | a / b – это деление нацело. Ответом будет два. Остаток от деления просто отбрасывается. |
| int a = 13;  int b = 5;  int d = a % b; // d == 3 | В d будет храниться остаток от деления a на b нацело. Остаток равен 3. |

— Но есть и пара интересных нюансов, которые следует помнить.

— Во-первых, ссылочная переменная не всегда хранит значение такого же типа, как и она.

— Во-вторых, при взаимодействии переменных двух разных типов, они должны быть сначала преобразованы к одному общему типу.

— А как же деление? Если разделить 1 на 3, мы получим 0.333(3). Ведь так?

— Нет не так. При делении двух целых чисел, результатом тоже будет целое число. Если разделить 5 на 3, то ответом будет 1 и два в остатке. Остаток при этом отбрасывается.

— Если разделить 1 на 3, то мы получим 0 (и единицу в остатке, который отбросится).

— А что делать, если я все же хочу получить 0.333 ?

— В Java числа перед делением лучше всего приводить к вещественному (дробному) типу путем умножения их на вещественную единицу – 1.0

|  |  |
| --- | --- |
| **Код на Java** | **Описание** |
| int a = 1/3; | a будет содержать 0 |
| double d = 1/3; | d будет содержать 0.0 |
| double d = 1.0 / 3; | d будет содержать 0.333(3) |
| double d = 1 / 3.0; | d будет содержать 0.333(3) |
| int a=5, b=7;  double d = (a\*1.0) / b; | d будет содержать 0.7142857142857143 |

— Понятно.