Пишем простое приложение

В этом видео мы напишем с вами простое приложение на языке Java. Это приложение будет эмулировать работу бензоколонки, а именно системы, которая рассчитывает общую стоимость заправки, исходя из типа бензина и количества литров.

Создание проекта

Для начала давайте создадим новый проект - так, как мы делали в предыдущем видео

Имя проекту зададим такое, чтобы оно соответствовало сути разрабатываемой программы - FillingStation

С английского “filling station” переводится как автозаправочная станция, или бензоколонка, если говорить проще.

Создадим в этом проекте сразу класс “Main” с методом “main()”.

Напишем первую строку:

```
"System.out.println("Система расчёта стоимости топлива");
```

Запустим, и проверим, что всё работает.

Именование переменных

Работоспособность и поддерживанемость программы

Можно назвать переменную просто “a”, “n” или “x”.

Но в Java принято давать переменным понятные имена, чтобы без комментариев было понятно, какая информация в ней хранится

Есть два фактора качества программного обеспечения:

- Работоспособность
- Поддерживаемость

Первый фактор — работоспособность: программа должна работать без ошибок, и выполнять ту задачу, которая перед стоит.

Второй фактор — это поддерживаемость: программа должна быть написана так, чтобы её можно было легко поддерживать, вносить в неё изменения, дорабатывать и исправлять ошибки.

И один из важнейших компонентов поддерживаемости — это читаемость, которая включает в себя правильный и понятный нейминг — именование всего, что есть в вашей программе.

Переменные должны быть названы понятно, и их названия должны отражать то, что в них содержится.

Когда-нибудь ваш проект станет очень большим, и если переменные в нём будут названы непонятно - вы не сможете в нём разобраться.

Кроме того, большинство программистов работают в командах, и там важно, чтобы не только вы сами, но и ваши коллеги понимали код друг друга. Поэтому код должен быть написан максимально понятно и доступно.

Вернёмся к нашему коду

В языке Java переменные принято называть с маленькой буквы, и если имя переменной состоит из нескольких слов, второе и каждое последующее слово пишется с большой буквы, а все эти слова пишутся слитно.

После имени переменной у нас идёт знак равно — это оператор присвоения значения: после него пишется значение, которое мы записываем в эту переменную.

Знак равно отделён пробелами. Этого можно не делать, но принято эти пробелы для читаемости кода.

Это похоже на переменные в алгебре: “x” равно 5, “y” равно 6, чему равно “z”? А у нас здесь просто переменные названы понятнее: “fuelType” равен 95.

В конце строки написана точка с запятой. На точку с запятой в Java должны оканчиваться все основные строки программного кода.

Мы можем подставить имя этой переменной в конструкцию System.out.ptintln и вывести в консоль, и при запуске программы в консоль будет выведено число, которое в эту переменную записано:

```
System.out.println(fuelType);
```

На вход наша система будет получать две переменные — тип топлива, которую мы уже задали, и количество литров.

Переменная — это место, где хранятся какие-то данные.

В первой переменной мы будем хранить тип бензина и будем обозначать его цифрой:

```
int fuelType = 95;
```

int — это означает, что переменная будет содержать в себе целые числа.
fuelType - имя переменной.

Для количества литров создадим вторую переменную:

```
int amount = 50;
```

Теперь давайте зададим цены для двух типов бензина. Цены бензина — это обычно и рубли, и копейки, поэтому используем переменные для хранения дробных величин:

```
double fuel92price = 60.2;
double fuel95price = 67.33;
```

И теперь напишем код, который будет выбирать нужную цену в зависимости от переданного в программу типа топлива:

```
double fuelPrice = 0;

if(fuelType == 92) {
      fuelPrice = fuel92price;
}
```

Двойное равно — это символ сравнения, и поэтому мы его используем в круглых скобках оператора “if”. А одинарное равно — это символ присвоения значения.

Напишем для второго типа топлива:

```
if(fuelType == 95) {
	fuelPrice = fuel95price;
}
```

И выведем в консоль выбранную цену:

```
System.out.println("Цена выбранного топлива: " + fuelPrice + " руб.");
```

Запускаем, смотрим, убеждаемся, что всё работает

Теперь давайте напишем код, который будет рассчитывать стоимость заправки, исходя из переданных параметров:

```
double totalPrice = fuelPrice * amount;
System.out.println("Общая стоимость заправки: " + totalPrice + " руб.");
```

Снова запускаем, смотрим, убеждаемся, что всё работает

Обработка ошибок

Первая ситуация — нам могут передать неверный тип топлива. В этом случае переменная “fuelPrice” так и останется равной нулю.

Давайте предусмотрим эту ситуацию:

```
if(fuelPrice > 0) {

} else {
	System.out.println("Выбран неверный тип топлива");
}
```

Давайте предусмотрим ситуацию, когда указано неверное количество топливо:

```
if(amount < 1) {
	System.out.println("Указано слишком малое количество топлива");
}
```

Предусмотрим ситуацию, при которой количество топлива будет отрицательным, потому что в этом случае цена тоже будет отрицательной. Сделаем просто:

```
if(amount < 1) {
      System.out.println("Указано слишком малое количество топлива");
      amount = 0;
}
```

И теперь, какими мы бы не были входные значения, код будет работать верно, даже если мы подставим некорректные значения.

Итоги

Мы написали программу для расчёта стоимости заправки на АЗС в зависимости от типа и количества топлива. В следующем уроке мы научимся упаковывать программы в jar-файлы для запуска на других компьютерах