Лекция 5

Типы данных

1. Основные сведения о типах данных
2. Примитивные типы
3. Ссылочные типы

1. Основные сведения о типах данных

Тип данных — фундаментальное понятие теории программирования.

Тип данных определяет множество значений, набор операций, которые можно применять к таким значениям, и, возможно, способ реализации хранения значений и выполнения операций.

Любые данные, которыми оперируют программы, относятся к определённым типам.

1.1. Иерархия типов данных

Java - строго типизированный язык

- Каждая переменная обладает типом
- Каждое выражение имеет тип
- Любые несоответствия типов являются ошибками, которые обязательно должны быть исправлены

Все типы данных разделяются на две группы:

- **примитивные типы данных (primitive types);**
- **ссылочные типы данных(reference types),**

1.1. Иерархия типов данных

Примитивные типы данных

Примитивные типы данных хранят значение.

а) Числовые типы
   - Целые типы (byte, short, int, long, char)
   - Вещественные типы (float, double)

б) Логический тип
   - boolean

1.1. Иерархия типов данных

Ссылочные типы

Ссылочные типы данных хранят ссылку на значение или набор значений.

- Классы
- Интерфейсы
- Массивы

1.2. Переменные

Переменные используются в программе для хранения данных. Любая переменная имеет *три* базовых характеристики:

- **имя;**
- **тип;**
- **значение.**

Имя уникально идентифицирует переменную и позволяет обращаться к ней в программе.

Тип описывает, какие величины может хранить переменная.

Значение – текущая величина, хранящаяся в переменной на данный момент.

1.2. Переменные

Обязательное объявление переменных в программе и возможная их инициализация при объявлении описываются следующим образом:

1. **Указывается тип переменной**
2. **Указывается имя переменной (можно через запятые указать имена нескольких переменных – они будут такого же типа)**
3. **Может указываться инициализатор (знак «присвоить» (=) с константой или выражением, вычисляемым во время компиляции или исполнения программы)**
4. **В конце записи ставится точка с запятой (;)**

1.2. Переменные

Примеры объявления переменных:

```
int a;
int b = 0, c = 3+2;
int d = b+c;
int e = a = 5;
```

После объявления переменная может применяться в различных выражениях, в которых будет браться ее текущее значение.

Кроме того, в любой момент можно изменить значение переменной, используя оператор присваивания, примерно так же, как это делалось в инициализаторах.

1.2. Переменные

При объявлении переменной может быть использовано ключевое слово final. Его указывают перед типом переменной, и тогда ее необходимо сразу инициализировать и уже больше никогда не менять ее значение. Таким образом, final-переменные становятся чем-то вроде констант.

Пример объявления final-переменной:

```
final double pi=3.1415;
```

2. Примитивные типы

Группа *примитивных* (*элементарных* или *простых*) типов данных подразделяется на три подгруппы:

1. **Целочисленные: byte, short, int, long, char**
2. **Дробные(вещественные): float, double**
3. **Логические(булевы): boolean**

2.1. Целочисленные типы

Таблица 5.1

Название типа	Длина (байты)	Область значений
byte	1	-128 .. 127
short	2	-32.768 .. 32.767
int	4	-2.147.483.648 .. 2.147.483.647
long	8	-9.223.372.036.854.775.808 .. 9.223.372.036.854.775.807 (примерно 10^{19})
char	2	'\u0000' .. '\uffff', или 0 .. 65.535

2.2. Дробные(вещественные) типы

Таблица 5.2

Название типа	Длина (байты)	Область значений
float	4	3.40282347e+38f ; 1.40239846e-45f
double	8	1.79769313486231570e+308 ; 4.94065645841246544e-324

2.3. Логический тип

Логический (булев) тип boolean может хранить всего два возможных значения:

- **true (истинно)**
- **false (ложно).**

Величины этого типа получаются в результате выполнения операций сравнения и логических операций.

2.4. Оболочечные классы

В ряде случаев вместо значения примитивного типа требуется объект. Например, для работы со списками объектов. Это связано с тем, что работа с объектами в Java может быть унифицирована, поскольку все классы Java являются наследниками класса Object, а для примитивных типов этого сделать нельзя.

Для таких целей в Java каждому примитивному типу сопоставляется объектный тип, то есть класс. Такие классы называются оболочечными (class wrappers).

Перечень оболочечных классов дан в табл. 5.3.

2.4. Оболочечные классы

Таблица 5.3

Примитивный тип	Оболочечный класс
byte	Byte
short	Short
char	Character
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean

2.4. Оболочечные классы

Классы Byte, Short, Integer, Long, Character содержат многие полезные методы для работы с целочисленными значениями, а классы Float и Double - для работы с дробными значениями Например, преобразование из текста в число.

Кроме того, есть класс Math, который хоть и предназначен в основном для работы с дробными числами, но также предоставляет некоторые возможности и для целых.

Класс Boolean обеспечивает много методов для преобразования boolean в String и обратно, а также другие константы и методы полезные при работе с логическим типом.

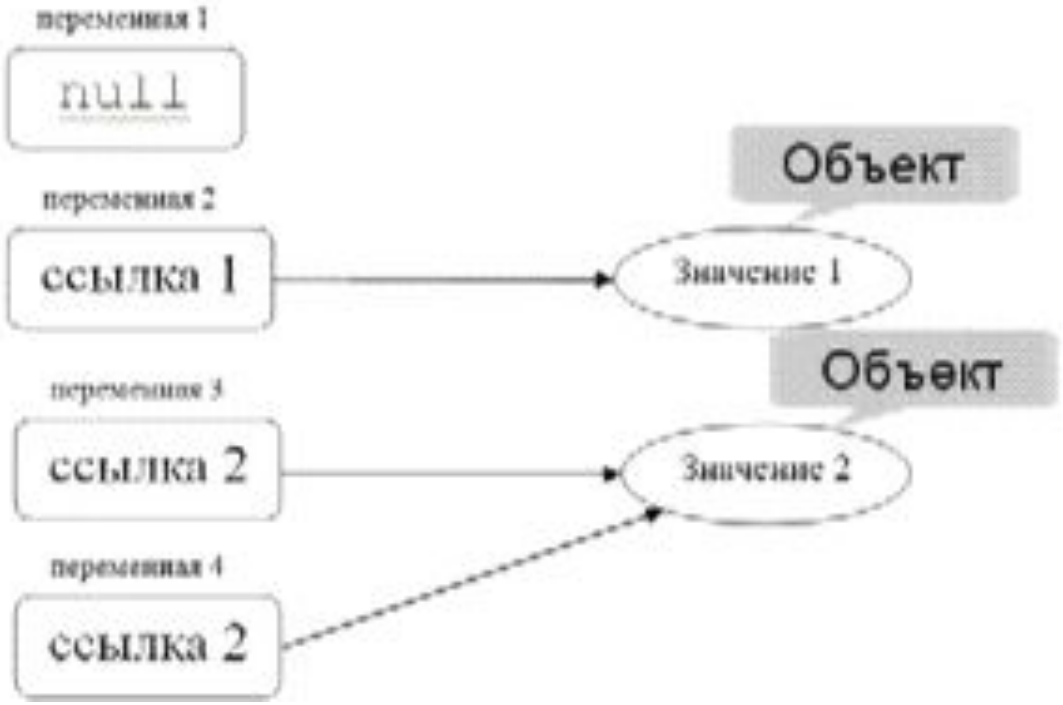
3. Ссылочные типы

Ссылочные типы данных хранят ссылку на значение или набор значений. К ним относятся все:

- **классы**
- **интерфейсы**
- **массивы**

3. Ссылочные типы

Различные ссылочные переменные могут ссылаться на один и тот же объект.



3.1. Объекты и правила работы с ними

Объект (object) – это экземпляр некоторого класса.

Класс – это описание объектов одинаковой структуры, и если в программе такой класс используется, то описание присутствует в единственном экземпляре.

```
// объявление класса Point
class Point {
   int x, y;
   Point (int newx, int newy){
      x=newx;
      y=newy;
   }
}
```

3.1. Объекты и правила работы с ними

Объектов этого класса может не быть вовсе, а может быть создано сколь угодно много.

```
Point p=new Point(1,2);
  // Создали объект, получили на него ссылку
Point p1=p;
  // теперь есть 2 ссылки на точку (1,2)
p=new Point(3,4);
  // осталась одна ссылка на точку (1,2)
p1=null;
```

Любой объект порождается только с применением ключевого слова new. Единственное исключение – экземпляры класса String. Записывая любой строковый литерал, мы автоматически порождаем объект этого класса.

3.1. Объекты и правила работы с ними

Над ссылочными значениями можно производить следующие операции:

- **обращение к полям и методам объекта**
- **оператор instanceof (возвращает булево значение)**
- **операции сравнения == и !=**
  (возвращают булево значение)
- **оператор приведения типов**
- **оператор с условием ?:**
- **оператор конкатенации со строкой +**

3.2. Класс Object

Каждый класс в Java может иметь только одного родителя. Таким образом, можно проследить цепочку наследования от любого класса, поднимаясь все выше. Существует класс, на котором такая цепочка всегда заканчивается, это класс Object. Именно от него наследуются все классы, в объявлении которых явно не указан другой родительский класс. А значит, любой класс напрямую, или через своих родителей, является наследником Object.

Отсюда следует, что методы этого класса есть у любого объекта (поля в Object отсутствуют), а потому они представляют особенный интерес. Основными из них являются:

- **getClass()**
- **equals()**
- **hashCode()**
- **toString()**
- **finalize()**

3.3. Класс String

Класс String занимает в Java особое положение. Экземпляры только этого класса можно создавать без ключевого слова new.

Каждый строковый литерал порождает экземпляр String, и это единственный литерал (кроме null), имеющий объектный тип.

Значение любого типа может быть приведено к строке с помощью оператора конкатенации строк.

Еще одним важным свойством данного класса является неизменяемость. Это означает, что, породив объект, содержащий некое значение-строку, уже нельзя изменить данное значение – необходимо создать новый объект.

В классе String определен метод intern(), который возвращает один и тот же объект-строку для всех экземпляров, равных по значению.

3.4. Класс Class

Класс Class является метаклассом для всех классов Java. Когда JVM загружает файл .class, который описывает некоторый тип, в памяти создается объект класса Class, который будет хранить это описание.

Например, если в программе есть строка

```
Point p=new Point(1,2);
```

то это означает, что в системе созданы следующие объекты:

1. Объект типа Point, описывающий точку (1,2) ;

2. Объект класса Class, описывающий класс Point ;

3. Объект класса Class, описывающий класс Object. Поскольку класс Point наследуется от Object, его описание необходимо;

4.Объект класса Class, описывающий класс Class. Это обычный класс, который должен быть загружен по общим правилам.

Выводы

1. Java является строго типизированным языком, то есть тип всех переменных и выражений определяется уже компилятором. Это позволяет существенно повысить надежность и качество кода, а также делает необходимым понимание программистами объектной модели.
2. Все типы в Java делятся на две группы – элементарные простые, или примитивные, типы (8 типов) и многочисленная группа ссылочных(объектных) типов (классов).
3. Примитивные переменные действительно являются хранилищами данных своего типа.
4. Ссылочные переменные хранят ссылку на некоторый объект совместимого типа. Они также могут принимать значение null, не указывая ни на какой объект. JVM подсчитывает количество ссылок на каждый объект и активизирует механизм автоматической сборки мусора для удаления неиспользуемых объектов.

Выводы

5. **Переменные имеют три основных параметра – имя, тип и значение. Любая переменная должна быть объявлена и при этом может быть инициализирована. Возможно использование модификатора final.**

6. **Примитивные типы состоят из пяти целочисленных, включая символьный тип, двух дробных и одного булевого.**

   Целочисленные литералы имеют ограничения, связанные с типами данных.

7. **Самые главные классы в Java – это классы Object, Class, String.**