Обзор Java

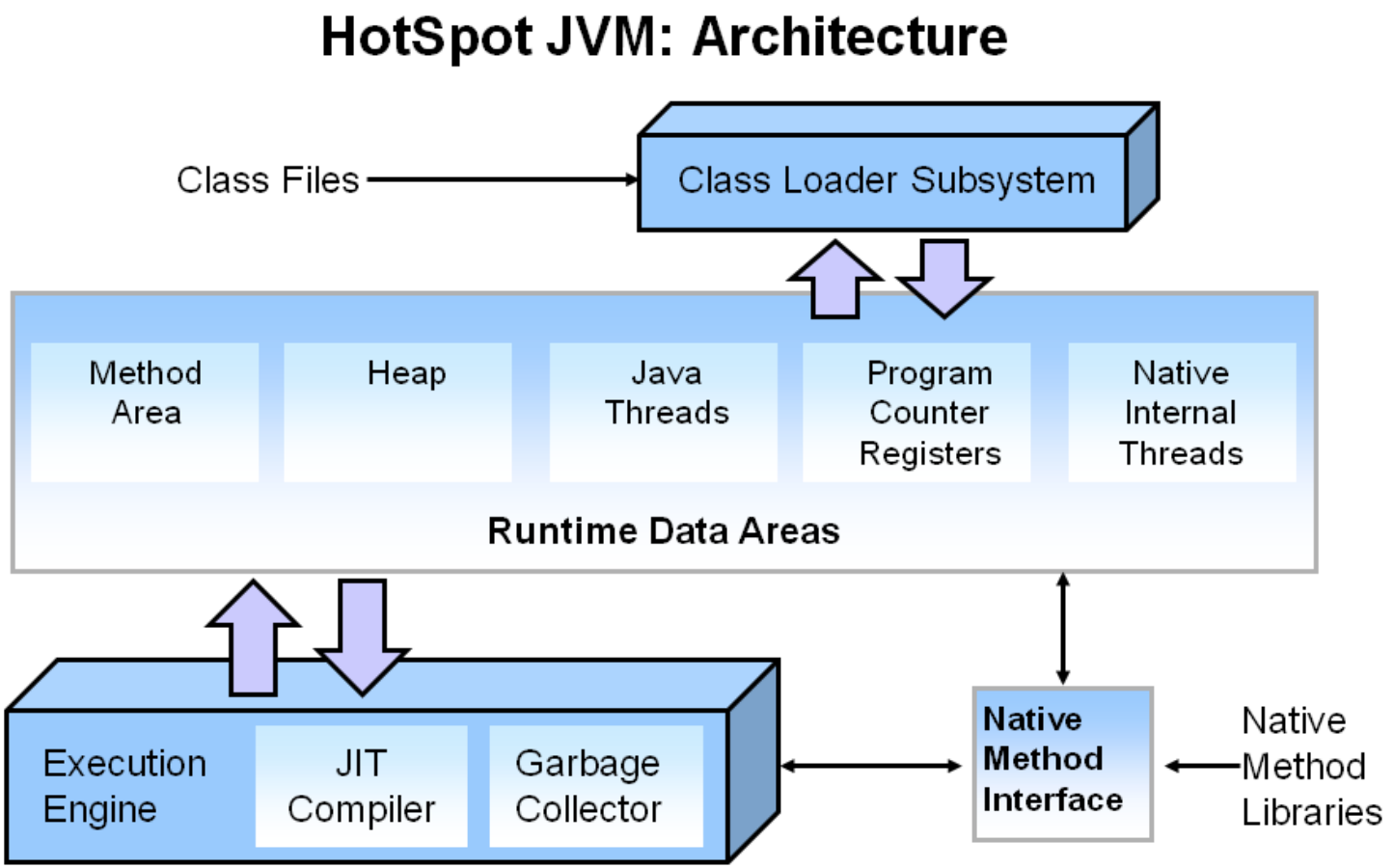
Занятие в ФМЛ 5
г. Долгопрудного.

Java

- Java (javac) компилятор компилирует в байт код.
- Байт код исполняется на виртуальной машине.
- НО! в процессе исполнения происходит компиляция, что позволяет достичь высокой производительности.

Java Virtual Machine

- © Oracle
-



Другие языки на виртуальной машине Java

- Scala
- Jython
- Много их

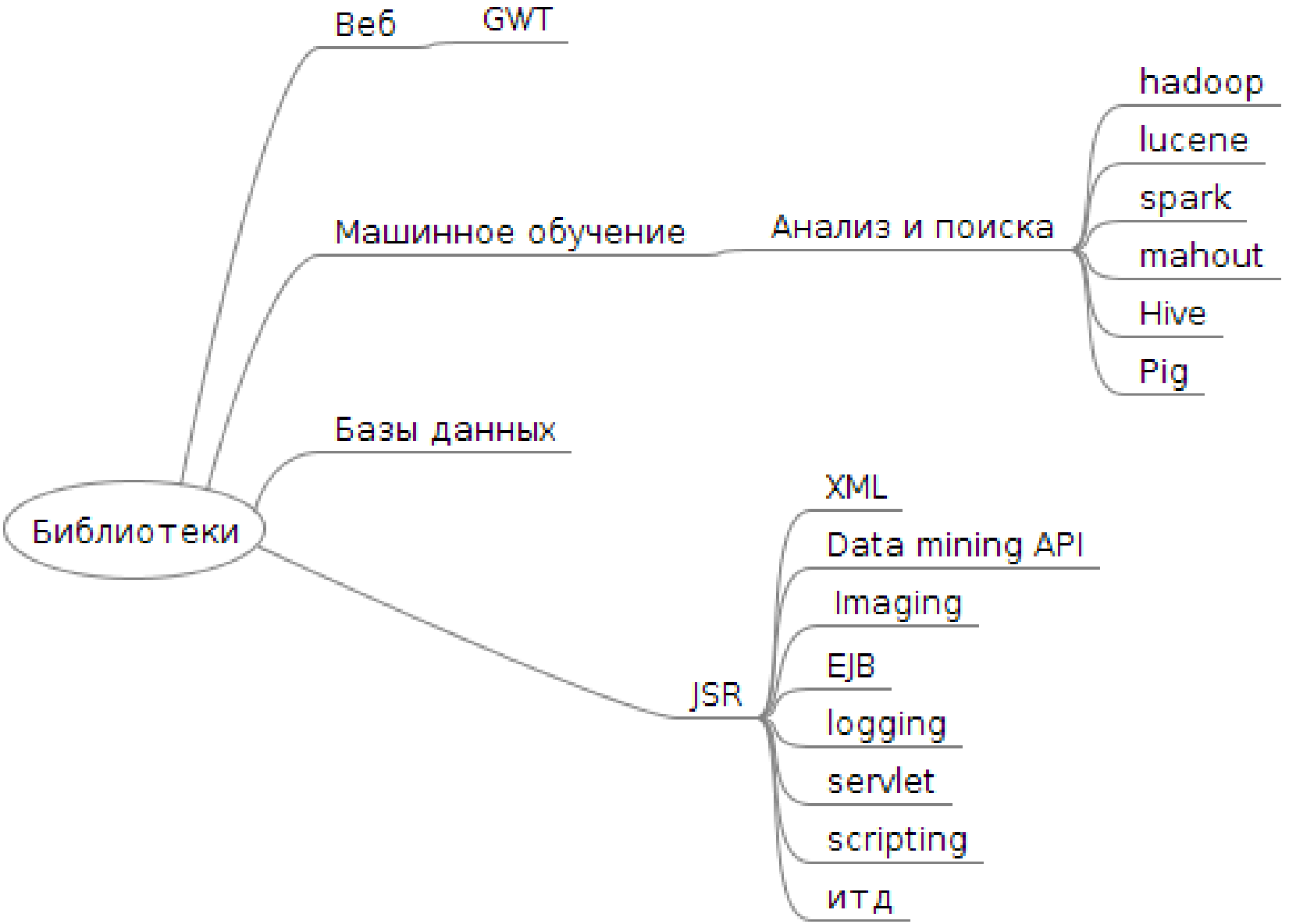
Что дает программисту

- Garbage collector
  - За:
    - Не нужно думать о памяти
      - Против: Ты не можешь контролировать память вручную, ее контролирует виртуальная машина, которую надо настраивать.
      - С другой стороны с контролем памяти в операционной системе (при вызове free/delete) тоже на практике бывает не все так просто.

Полностью объектно-ориентированный.

- Все есть класс, даже функция main может быть только внутри класса
- Generics — не шаблоны

Накоплено много библиотек



Продвинутые среды разработки и рефакторинга

- Eclipse
- Netbeans
- Intelij IDEA

- Система плагинов

Проприетарная

- Принадлежит ORACLE
- Хотя есть реализации не JVM не ORACLE — на практике с ними больше проблем, в силу меньшей распространенности

Java vs C++

- Памяти требует больше
- Среды лучше и скрость разработки
- Скорость выполнения в целом меньше
- C++ язык для высокопроизводительных вычислений, Java — для enterpise.
-

Пример программы

```
public class Sequence {
    Sequence() { System.out.print("c "); }
    { System.out.print("y "); }
    public static void main(String[] args) {
        System.out.print("m ");
        new Sequence().go();
    }
    void go() { System.out.print("g "); }
    static { System.out.print("x "); }
}
```

Вывод

```
x m y c g
```

Vector

```
static void display(int X) {
System.out.println(X);
}

    public static void main(String[] args) {
        Vector<Integer> vec = new Vector<>();

        vec.addAll(Arrays.asList(1,2,3,4,5,6,7,8,9,10));
        vec.add(5,105);
        vec.add(11);

        Consumer<Integer> func = x->display(x);

        vec.forEach(func);
}
```

Map

```
Map<String, Integer> map = new TreeMap<>();

map.put("One", 1);
map.put("Two", 2);
map.put("Three",3);

map.put("Four",7);

map.replace("Four", 4);

BiConsumer<String,Integer> funmap = (x,y)->{
    System.out.println(x+"->"+y);
    };

map.forEach(funmap);
```

Сложные объекты

- Vector<Map<String,Integer> > vecmapstrint =
- new Vector<Map<String, Integer>>();
-
- for(int a=0; a<100;a++){
- vecmapstrint.add(map);
- };

Array, что будет выведено?

- int a[][] = {{1,2,3,4},{1,2,3}};
-
- System.out.println(a);
-
-

Array

- [[I@18025c

Проверка, что выведется?

```
int a[][] = {{1,2,3,4},{1,2,3},{},{9,9}};

System.out.println("a.size:" + a.length);

for (int[] is : a) {
    for (int i : is) {
        System.out.print(i + " ");
    }
    System.out.println( "; sz=" + is.length);
}
```

Проверка

- 1 2 3 4 ; sz=4
- 1 2 3 ; sz=3
- ; sz=0
- 9 9 ; sz=2

Практикум

- Попробуем установить и проделать на рабочем месте.

Вопросы

- Спасибо!