Main Method

Whether the class contains main() method or not, and whether it is properly declared or not, these checking's are not responsibilities of the compiler, at runtime JVM is responsible for these. If JVM unable to find the required main() method then we will get runtime exception saying NoSuchMethodError: main.

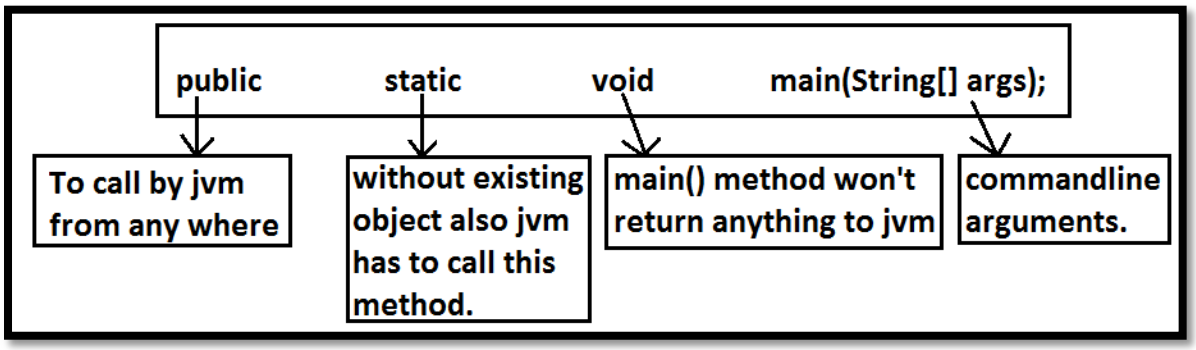
Example:

```
class Test
{}
```

Output:

```
javac Test.java
java Test R.E: NoSuchMethodError: main
```

At runtime JVM always searches for the main() method with the following prototype.



If we are performing any changes to the above syntax then the code won't run and will get Runtime exception saying NoSuchMethodError.

Even though above syntax is very strict but the following changes are acceptable to main() method.

The order of modifiers is not important that is instead of public static we can take static public.

We can declare string[] in any acceptable form
String[] args
String []args
String args[]
Instead of args we can use any valid java identifier.

We can replace string[] with var-arg parameter.

Example: main(String... args)
main() method can be declared with the following modifiers.
final, synchronized, strictfp.

```
1)  public class  Test
2)  {
3)     static final synchronized strictfp public void main(String... durga)
4)     {
5)        System.out.println("Valid Main Method");
6)     }
7)  }
```

Output: Valid Main Method

Q. Which of the following main() method declarations are valid ?

public static void main(String args){} (invalid)
public synchronized final strictfp void main(String[] args){} (invalid)
public static void Main(String... args){} (invalid)
public static int main(String[] args){} //int return type we can't take //(invalid)
public static synchronized final strictfp void main(String... args){}(valid)
public static void main(String... args){}(valid)
public void main(String[] args){}(invalid)

In which of the above cases we will get compile time error ?

No case, in all the cases we will get runtime exception.

Case 1 :
Overloading of the main() method is possible but JVM always calls string[] argument main() method only.

Example:

```
1)  class Test
2)  {
3)     public static void main(String[] args)
4)     {
5)     System.out.println("String[] array main method"); //overloaded methods
6)     }
7)     public static void main(int[] args)
8)     {
9)     System.out.println("int[] array main method");
10)    }
11) }
```


DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

Output:

String[] array main method
The other overloaded method we have to call explicitly then only it will be executed.

Case 2:

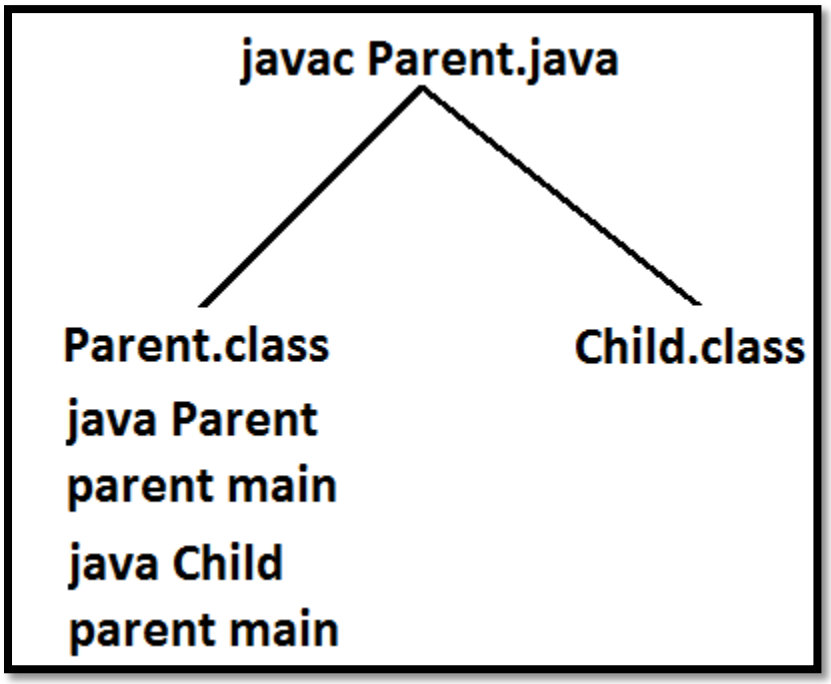
Inheritance concept is applicable for static methods including main() method
hence while executing child class if the child class doesn't contain main() method then the parent class main() method will be executed.

Example 1:

Parent.java

```
1)  class Parent
2)  {
3)     public static void main(String[] args)
4)     {
5)        System.out.println("parent main");          }
6)  }
7)  class Child extends Parent
8)  {
9)  }
```

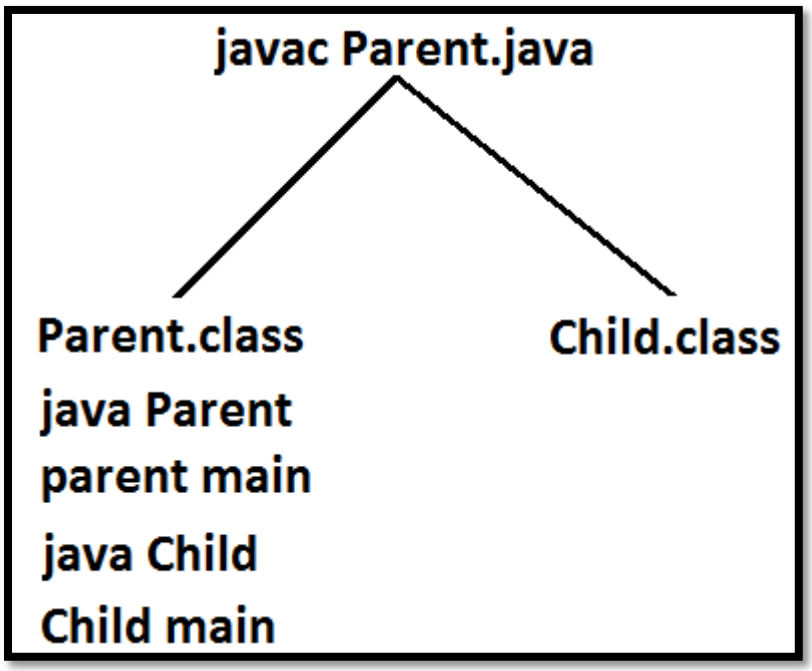
Analysis:



Example 2:

```
1)  class Parent
2)  {
3)     public static void main(String[] args)
4)     {
5)        System.out.println("parent main");       // Parent.java
6)     }
7)  }
8)  class Child extends Parent
9)  {
10)    public static void main(String[] args)
11)    {
12)       System.out.println("Child main");
13)    }
14) }
```

Analysis:



It seems to be overriding concept is applicable for static methods but it is not overriding it is method hiding.

1.7 Version Enhansements with respect to main():

Case 1 :

Untill 1.6v if our class doesn't contain main() method then at runtime we will get Runtime Exception saying NosuchMethodError:main
But from 1.7 version onwards instead of NoSuchMethodError we will get more meaning full description

```
class Test {
}
```

1.6 version:

```
javac Test.java
java Test
RE: NoSuchMethodError:main
```

1.7 version:

```
javac Test.java
java Test
Error: main method not found in class Test, please define the main  method as
public static void main(String[]  args)
```

Case 2:

From 1.7 version onwards to start program execution compulsory main method should be required, hence even though the class contains static block if main method not available then won't be executed

```
1)  class Test {
2)   static {
3)    System.out.println("static block");
4)   }
5)  }
```

1.6 version:

```
javac Test.java
java Test
output :
static block
RE: NoSuchMethodError:main
```


DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

1.7 version:

javac Test.java
java Test
Error: main method not found in class Test, please define the main method as
public static void main(String[] args)

Case 3:

```
1)  class Test {
2)   static {
3)    System.out.println("static block");
4)    System.exit(0);
5)   }
6)  }
```

1.6 version:

javac Test.java
java Test
output :
static block

1.7 version:

javac Test.java
java Test
Error: main method not found in class Test, please define the main method as
public static void main(String[] args)

Case 4:

```
1)  class Test {
2)   static {
3)    System.out.println("static block");
4)   }
5)   public static void main(String[] args) {
6)    System.out.println("main method");
7)   }
8)  }
```

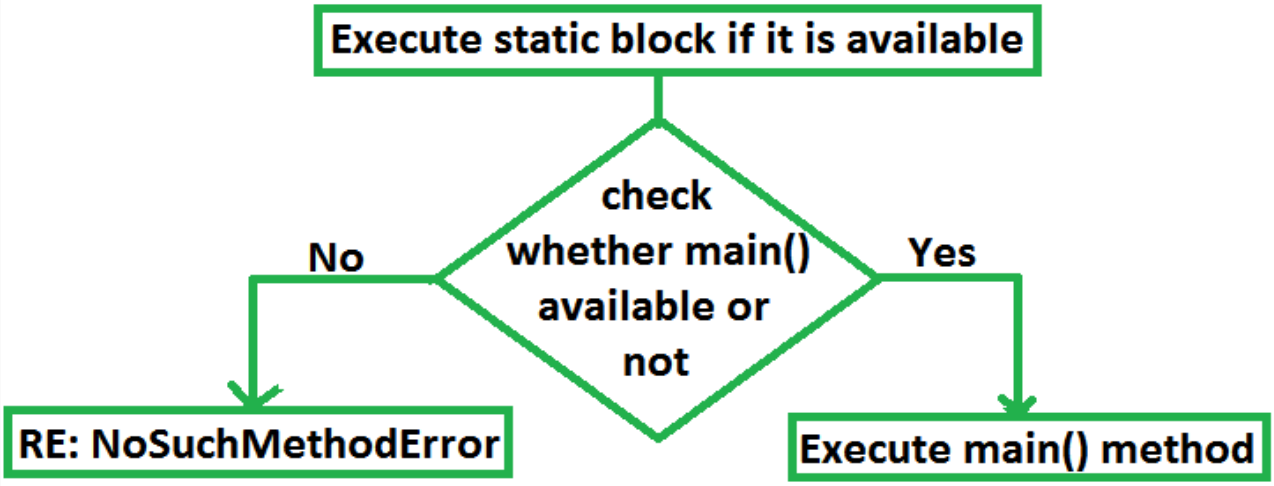
1.6 version:

javac Test.java
java Test
output :
static block
main method

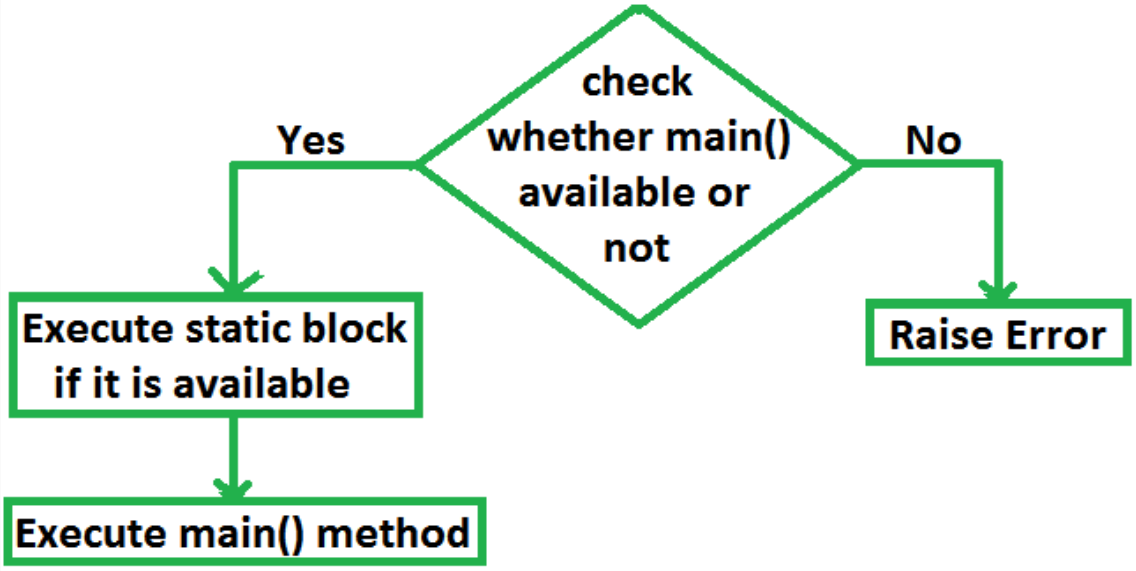
1.7 version:

javac Test.java
java Test
output :
static block
main method

1.6 version :



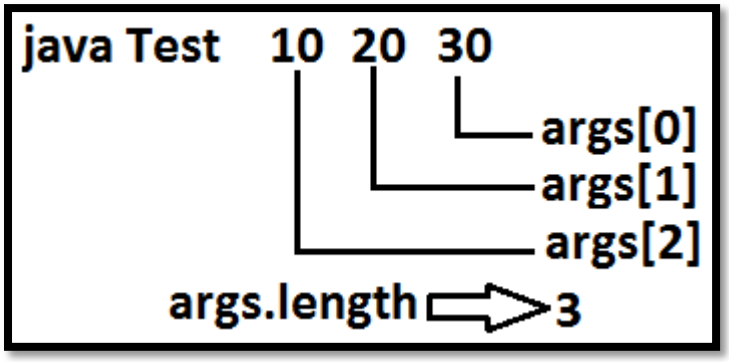
1.7 version :



DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

Command line arguments:

The arguments which are passing from command prompt are called command line arguments.
The main objective of command line arguments are we can customize the behavior of the main() method.



Example 1:

```
1)  class Test
2)  {
3)     public static void main(String[] args)
4)     {
5)        for(int i=0;i<=args.length;i++)
6)        {
7)           System.out.println(args[i]);
8)        }
9)     }
10) }
```

Output:
java Test x y z
ArrayIndexOutOfBoundsException: 3
Replace i<=args.length with i<args.length then it will run successfully.

Example 2 :

```
1)  class Test
2)  {
3)     public static void main(String[] args)
4)     {
5)      String[] argh={"X","Y","Z"};
6)       args=argh;
7)        for(String s : args)
8)        {
9)           System.out.println(s);
10)       }
```


DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

```
11)    }
12) }
```

Output:

```
java Test A B C
X
Y
Z

java Test A B
X
Y
Z

java Test
X
Y
Z
```

Within the main() method command line arguments are available in the form of String hence "+" operator acts as string concatenation but not arithmetic addition.

Example 3 :

```
1)  class Test
2)  {
3)     public static void main(String[] args)
4)     {
5)        System.out.println(args[0]+args[1]);
6)     }
7)  }
```

Output:

```
E:\SCJP>javac Test.java
E:\SCJP>java Test 10 20
1020
```

Space is the separator between 2 command line arguments and if our command line argument itself contains space then we should enclose with in double quotes.

Example 4 :

```
1)  class Test
2)  {
3)     public static void main(String[] args)
4)     {
5)        System.out.println(args[0]);
6)     }
7)  }
```

Output:

```
E:\SCJP>javac Test.java
E:\SCJP>java Test "Sai Charan"
Sai Charan
```

Q. Which one of the following code examples uses valid java syntax?

A)

```
1)  public class Bunny
2)  {
3)     public static void main(String[] args)
4)     {
5)        System.out.println("Bunny");
6)     }
7)  }
```

B)

```
1)  public class Chinny
2)  {
3)     public static void main(String[])
4)     {
5)        System.out.println("Chinny");
6)     }
7)  }
```

C)

```
1)  public class Sunny
2)  {
3)     public void main(String[] args)
4)     {
5)        System.out.println("Sunny");
6)     }
7)  }
```

D)

```
1)  public class Vinny
2)  {
3)     public static void main(String() args)
4)     {
5)        System.out.println("Vinny");
6)     }
7)  }
```

Answer: A

Q. Given the code from the Test.java file:

```
1)  public class Test
2)  {
3)     public static void main(String[] args)
4)     {
5)        System.out.println("Hello "+args[0]);
6)     }
7)  }
```

Which set of commands prints Hello Durga in the console?

A) javac Test
java Test Durga

B) javac Test.java Durga
java Test

C) javac Test.java
java Test Durga

D) javac Test.java
java Test.class Durga

Answer: C

Q. Consider the code Test.java:

```
1)  public class Test
2)  {
3)     public static void main(int[] args)
4)     {
5)        System.out.println("int[] main: "+args[0]);
6)     }
7)     public static void main(Object[] args)
8)     {
9)        System.out.println("Object[] main: "+args[0]);
10)    }
11)    public static void main(String[] args)
12)    {
13)       System.out.println("String[] main: "+args[0]);
14)    }
15) }
```

and the commands
javac Test.java
java Test 1 2 3

What is the result?

A) int[] main 1
B) Object[] main 1
C) String[] main 1
D) Compilation Fails
E) An Exception raises at runtime

Answer: C

DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com