Valid and Invalid `` Methods in Java

Valid `` Method Signatures ✅

The following `main` method signatures are **valid** in Java:

public static void main(String[] args)
public static void main(String []args)
public static void main(String args[])
public static void main(String... args)
static public void main(String[] args)
public static final void main(String[] args)
final public static void main(String[] args)
final strictfp public static void main(String[] args)

Why are they valid?

- The method is `public` (so JVM can access it from anywhere).
- The method is `static` (so JVM can call it without an instance of the class).
- The return type is `void` (since the `main` method does not return any value).
- The method takes a `String[] args` parameter (or equivalent variations like `String args[]`, `String []args`, `String... args`).
- The order of modifiers (`public static` vs. `static public`) does not affect the validity.

Invalid `` Method Signatures ❌

The following `main` method signatures are **invalid** in Java:

public void main(String[] args) // ❌ Missing 'static'
static void main(String[] args) // ❌ Missing 'public'
public void static main(String[] args) // ❌ Incorrect order of modifiers
abstract public static void main(String[] args) // ❌ 'abstract' is not allowed

Why are they invalid?

- `` → Missing `static`. The `main` method must be static because JVM calls it without creating an instance of the class.
- `` → Missing `public`. The JVM requires `main` to be `public` so it can be accessed from outside the class.
- `` → Incorrect order of modifiers. The correct order should be `public static`.

- `` → `abstract` methods cannot have a body, but `main` must have a definition to be executed.

Conclusion

When defining the `main` method in Java, always ensure that:

1. The method is `public static void`.
2. The parameter is `String[] args` (or equivalent variations).
3. Additional modifiers like `final`, `strictfp`, and reordered `public static` are allowed.
4. It does not have invalid modifiers like `abstract` or incorrect method signatures.

This ensures that your Java program runs correctly when executed by the JVM.