Chapter 18

static methods

static method

```
class Welcome{

 public static void main(String[] args){
    System.out.println("Welcome to suresh techs, I am learning Java.");
    System.out.println("My name is suresh, I will get job soon");
    System.out.println(1);
    System.out.println(2);
    System.out.println(3);
    System.out.println(4);
    System.out.println("\"Suresh techs\" is 5 star");
 }

}
```

Tell me one thing

- Can we **call a method of one class in another class?**
- Try to call **getRollno()** method from **StaticDemo** class
- We can call **getRollno()** method only by creating instance(object) of Student class
- But, **static methods can be called** from other classes **without creating objects**

```
        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);
        student1.marks = 90;
//      student1.totalStudents = 1;

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(student.college);
```

static method

- **Static methods** belong to a **class** rather than object
- So it **can be called** directly by using **classname.methodname**
- create a **static method** named **getTotalStudents()** and it should return **totalStudents** value

```
static int getTotalStudents(){
    return totalStudents;
}
```

So, there is no need to create an object to call a static method of a class from outside

That's why main method needs to be a static method as JVM doesn't need to create an object to call the method from outside – (java classname)

```
System.out.println("Total students: "+Student.getTotalStudents());
```

static method

Can be accessed from everywhere

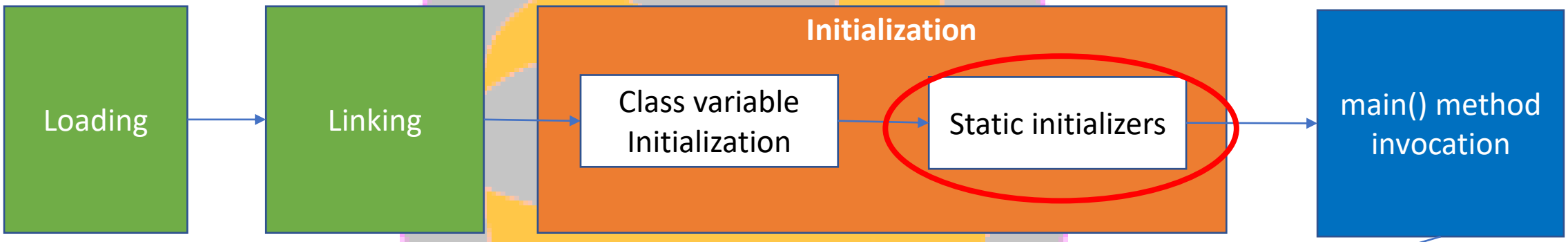
Don’t need to create an object

Doesn’t return anything

```
class Welcome{

 public static void main(String[] args){
     System.out.println("Welcome to suresh techs, I am learning Java.");
     System.out.println("My name is suresh, I will get job soon");
     System.out.println(1);
     System.out.println(2);
     System.out.println(3);
     System.out.println(4);
     System.out.println("\"Suresh techs\" is 5 star");
 }

}
```

Points to remember

- Only the **main() method** which is static will be called by the **JVM automatically**, **not all the static methods will be called automatically**



Static variables are also called class variables and class initialization will initialize static variables

```
class Welcome{

 public static void main(String[] args){
    System.out.println("Welcome to suresh techs, I am learning Java.");
    System.out.println("My name is suresh, I will get job soon");
    System.out.println(1);
    System.out.println(2);
    System.out.println(3);
    System.out.println(4);
    System.out.println("\"Suresh techs\" is 5 star");
 }

}
```

Points to remember – very important

- **Instance methods** can directly access **both instance methods and instance variables**
- **Instance methods** can also access **static variables and static methods directly**
- **Static methods can access all static variables and static methods directly**

TASK

👉 **Static methods can't access instance variables and instance methods directly. They need object reference to access.**

Task – Suresh Techs **Viewer**

- Create a class named **Viewer**
- Think of the state of Viewer
  - **Name of type String**
  - **IsLiked of type boolean**
  - **IsSubscribed of type of boolean**

```
class Viewer{
    String name;
    boolean isLiked;
    boolean isSubscribed;

    public static void main(String[] args){

    }
}
```

- Create **instance variables** for the **three states**

```
error: non-static variable name cannot be referenced from a static context
 System.out.println(name);
                    ^
```

- **Create main method**

- Print **value of name in the main method**

Task

- Create **object of the Viewer** and access name
- Set **name** of the viewer as **Suresh**
- Set **isLiked** to **true** and **isSubscribed** to false
- Display **name, isLiked, isSubscribed**
- Create a method called **thankYou() and that method should perform below operation**
    - **Thank you Suresh**
    - **If liked**
        - **Thank you for liking this video**
    - **If subscribed**
        - **Thank you subscribing our channel**
    - **If liked & subscribed**
        - **Thank you for liking and subscribing**

```java
class Viewer{
    String name;
    boolean isLiked;
    boolean isSubscribed;

    public static void main(String[] args){
        Viewer v1 = new Viewer();
        System.out.println("Name: "+v1.name);
    }
}
```

```java
public static void main(String[] args){
    Viewer v1 = new Viewer();
    v1.name = "Suresh";
    v1.isLiked = true;
    v1.isSubscribed = false;
    System.out.println("Name: "+v1.name);
    System.out.println("Liked: "+v1.isLiked);
    System.out.println("Subscribed: "+v1.isSubscribed);
}
```

```java
public static void main(String[] args){
    Viewer v1 = new Viewer();
    v1.name = "Suresh";
    v1.isLiked = true;
    v1.isSubscribed = false;
    System.out.println("Name: "+v1.name);
    System.out.println("Liked: "+v1.isLiked);
    System.out.println("Subscribed: "+v1.isSubscribed);
    thankYou();
}
```

```java
void thankYou(){
    System.out.println("Thank you "+name);
    if(isLiked && isSubscribed){
        System.out.println("Thank you for liking and subscribing");
    }else if(isLiked){
        System.out.println("Thank you for liking");
    }else if(isSubscribed){
        System.out.println("Thank you for subscribing");
    }
}
```

```
error: non-static method thankYou() cannot be referenced from a static context
thankYou();
^
```

Create a static method

- Create a static method called **wish()**

- Which should say **“Welcome to suresh techs, all the very best for your future. You will get JOB for sure”**

```
static void wish(){
    System.out.println("Welcome to suresh techs, all the very best for your future"
    +"You will get JOB for sure");
}
```

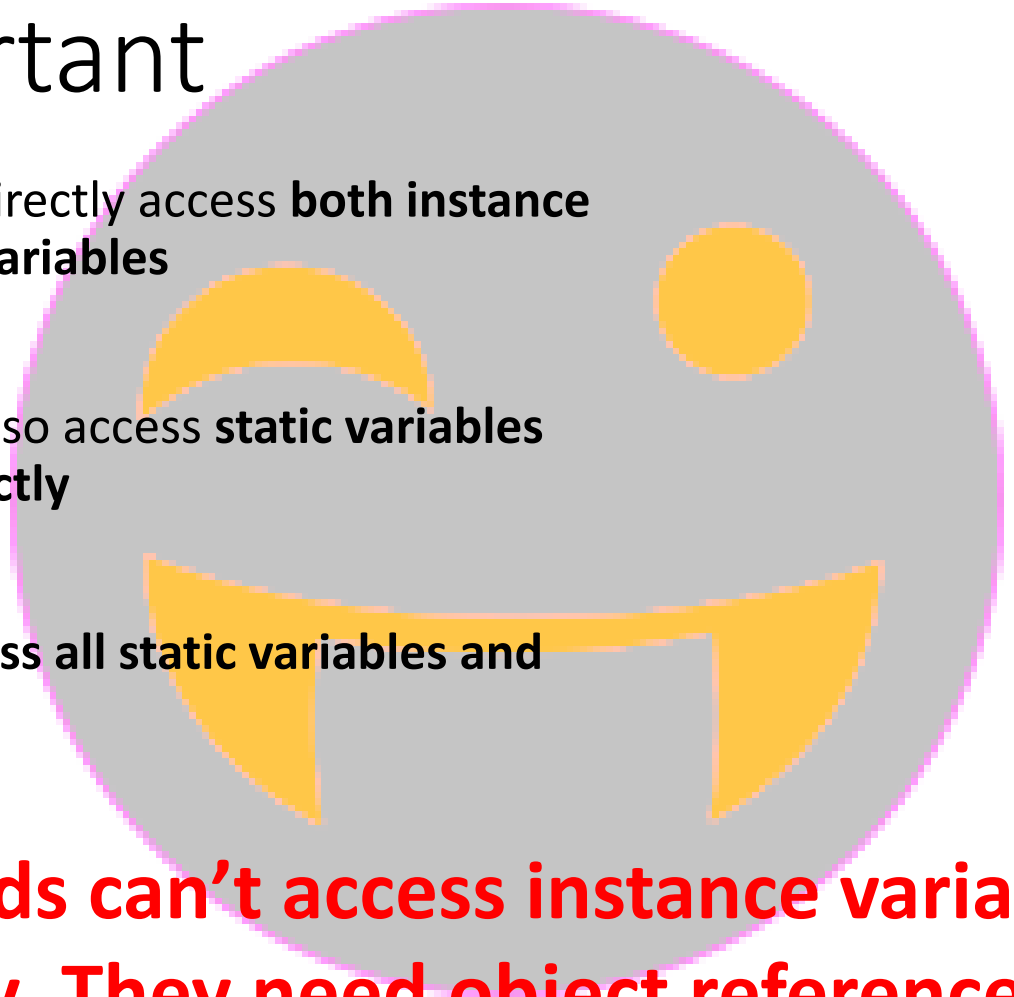
Call wish() method from thankYou() method

Calling **wish() method from thankYou() method**

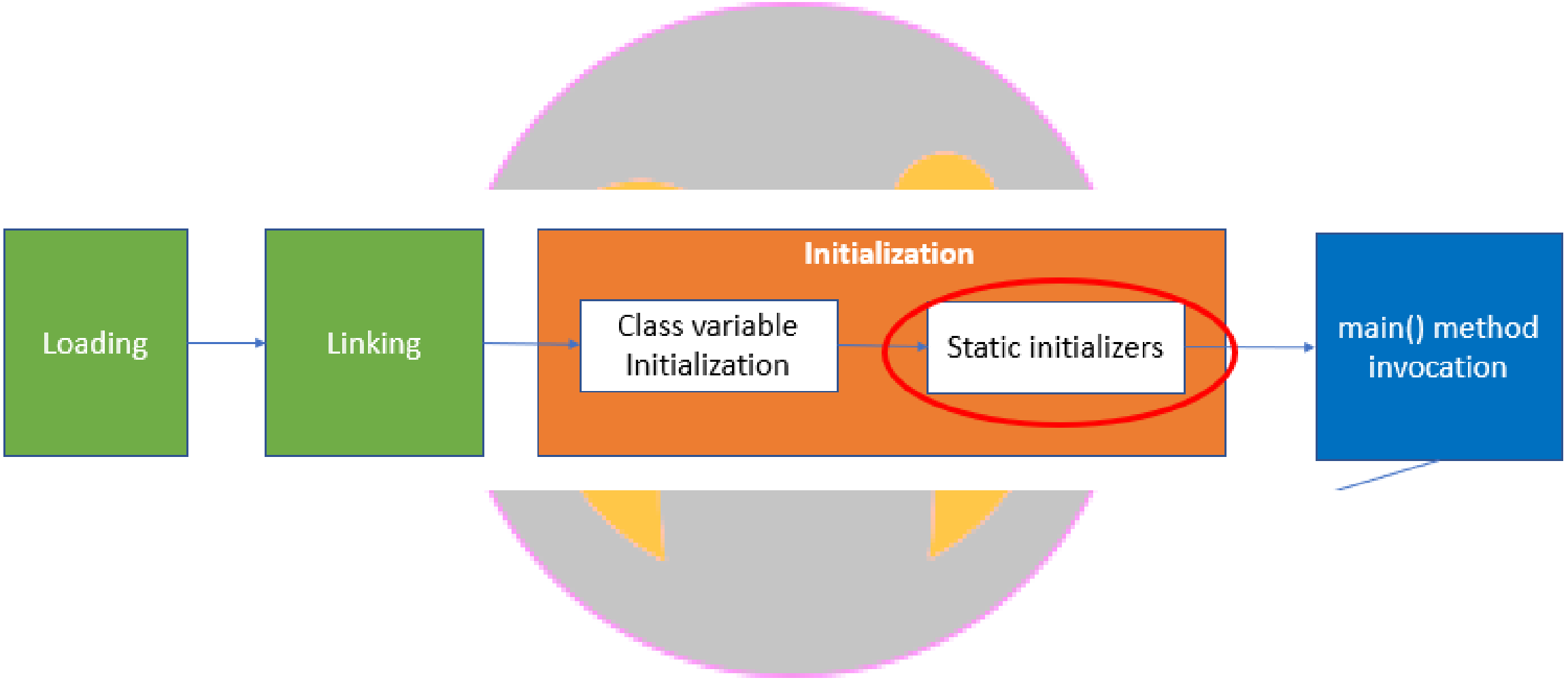
```
void thankYou(){
    System.out.println("Thank you "+name);
    if(isLiked && isSubscribed){
        System.out.println("Thank you for liking and subscribing");
    }else if(isLiked){
        System.out.println("Thank you for liking");
    }else if(isSubscribed){
        System.out.println("Thank you for subscribing");
    }
    wish();
}

static void wish(){
    System.out.println("Welcome to suresh techs, all the very best for your future"
    +"You will get JOB for sure");
}
```

Let’s look those important points – very important

- **Instance methods** can directly access **both instance methods and instance variables**
- **Instance methods** can also access **static variables and static methods directly**
- **Static methods can access all static variables and static methods directly**



👉 **Static methods can’t access instance variables and instance methods directly. They need object reference to access.**


Loading
Linking
Initialization
Class variable Initialization
Static initializers
main() method invocation

```java
public static void main(String[] args){
    System.out.println("Variables Demo");
    int a, b, c, d;
    a = 10;
    b = 20;
    c = -20;
```

Local variables

```java
class Student{
    String name;
    String studyClass;
    int rollno;
    double percentage;
    House h;
    static String college="Suresh Techs College";
    int marks;
    static int totalStudents;
```

Instance variables & instance methods

Static variables & static methods

Every **variable is assigned a data type** that describes the **type** and **quantity** of value it can hold

What next?

Data types in detail

చిన్న బ్రేక్ చిటికలో వచ్చేస్తా