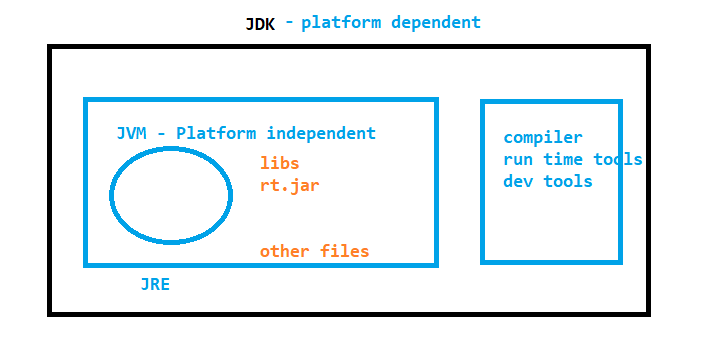
Java :



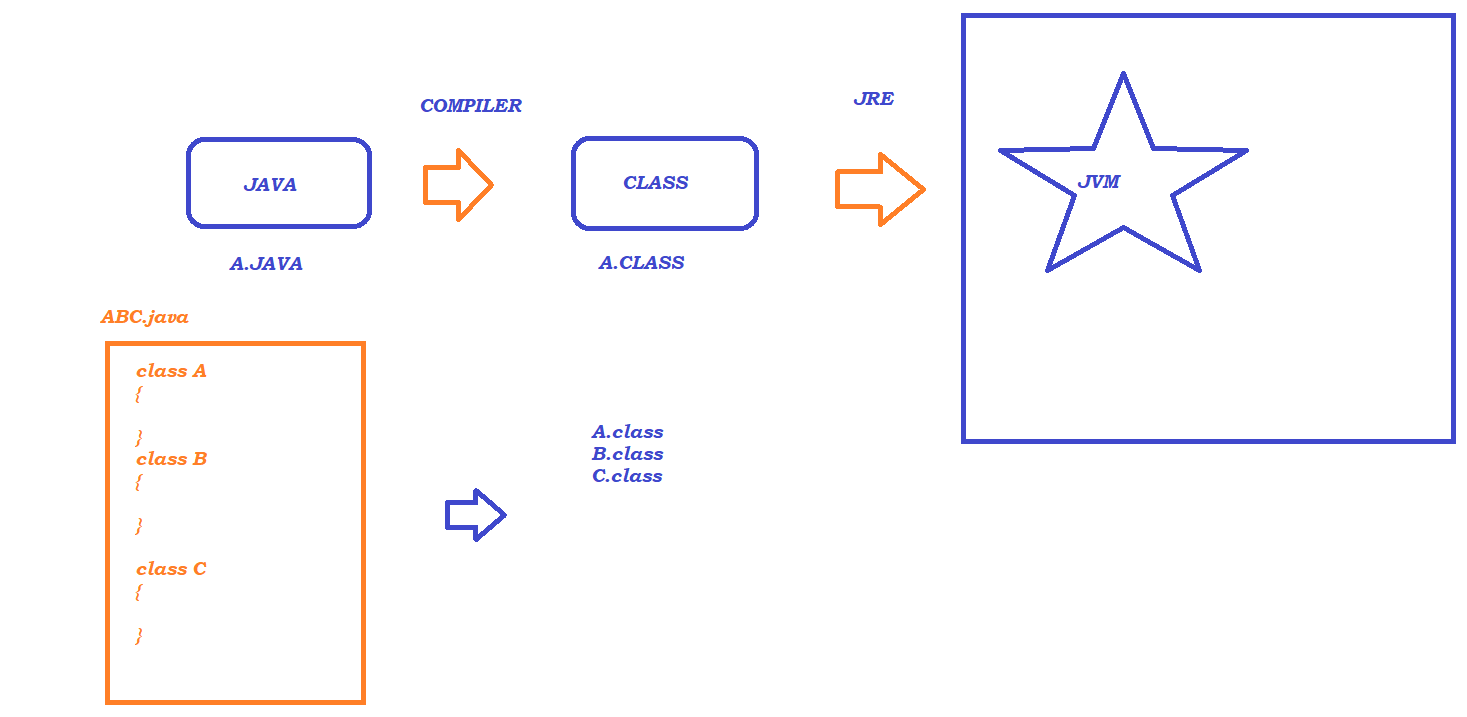
jdk :

jre :

jvm :

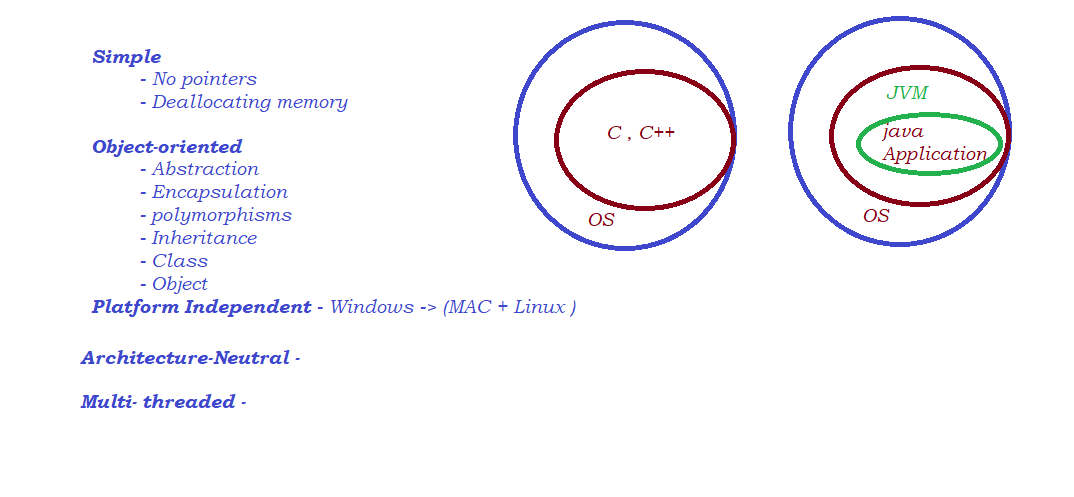
Execution –

1. extension should be .java
2. atleast one class should be present
3. Run a program : main method is mandatory
4. inside one file as many classes as we want
5. valid name to the file or class

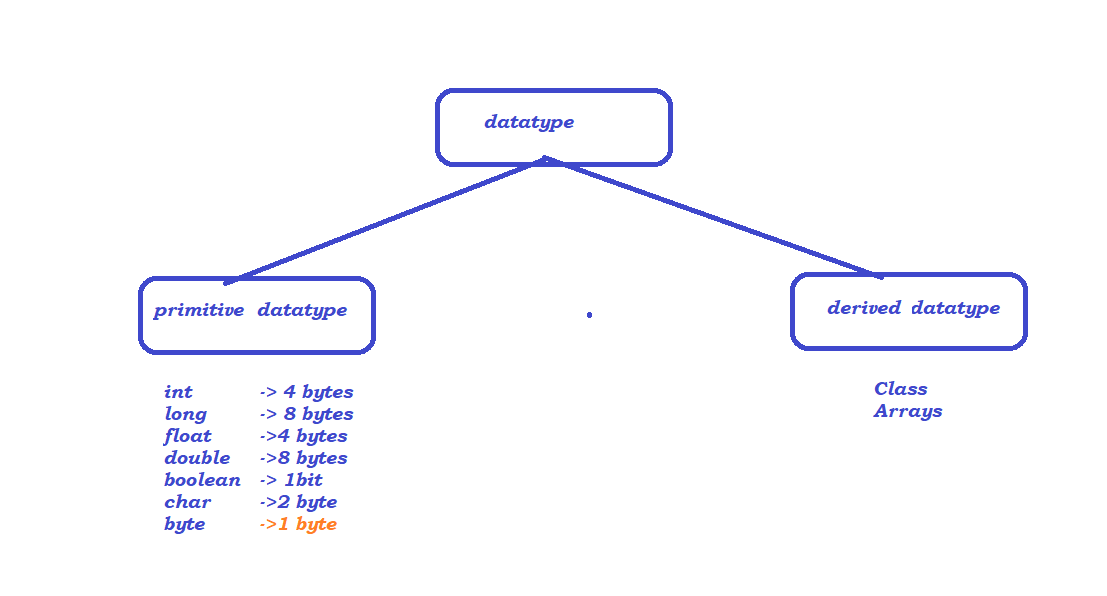


Features :

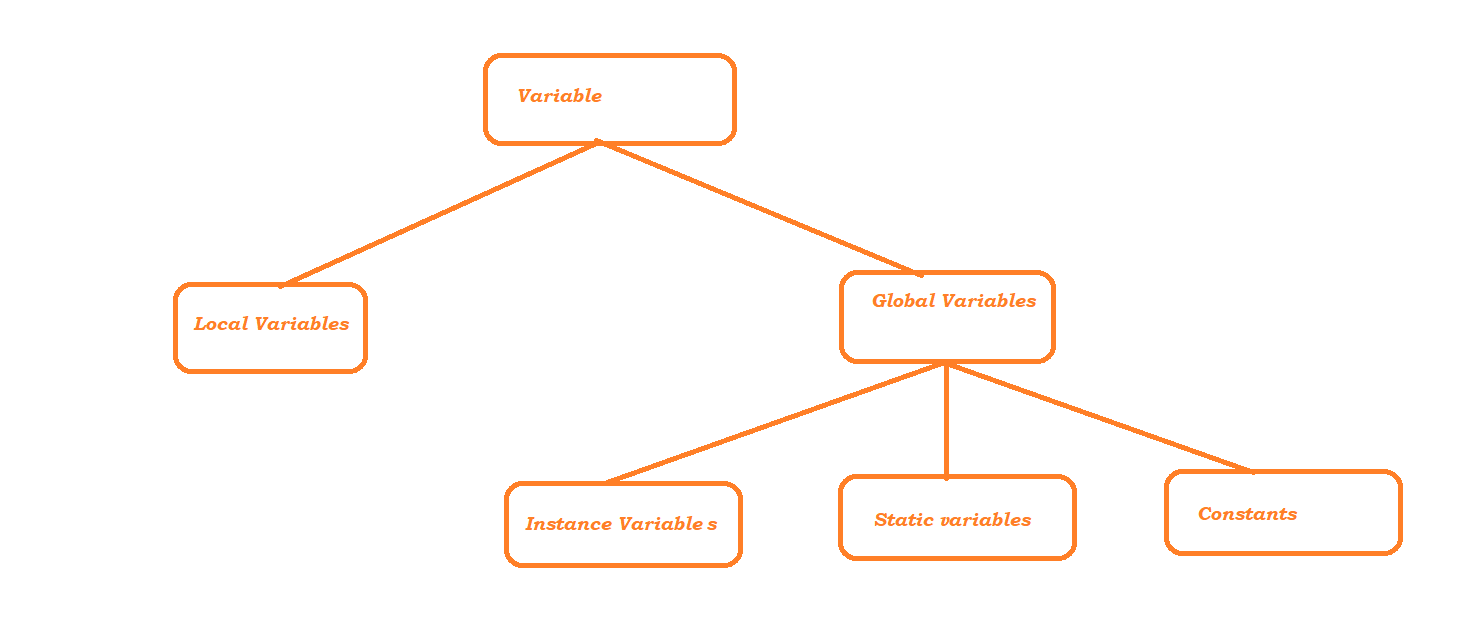
---------



Data Type :



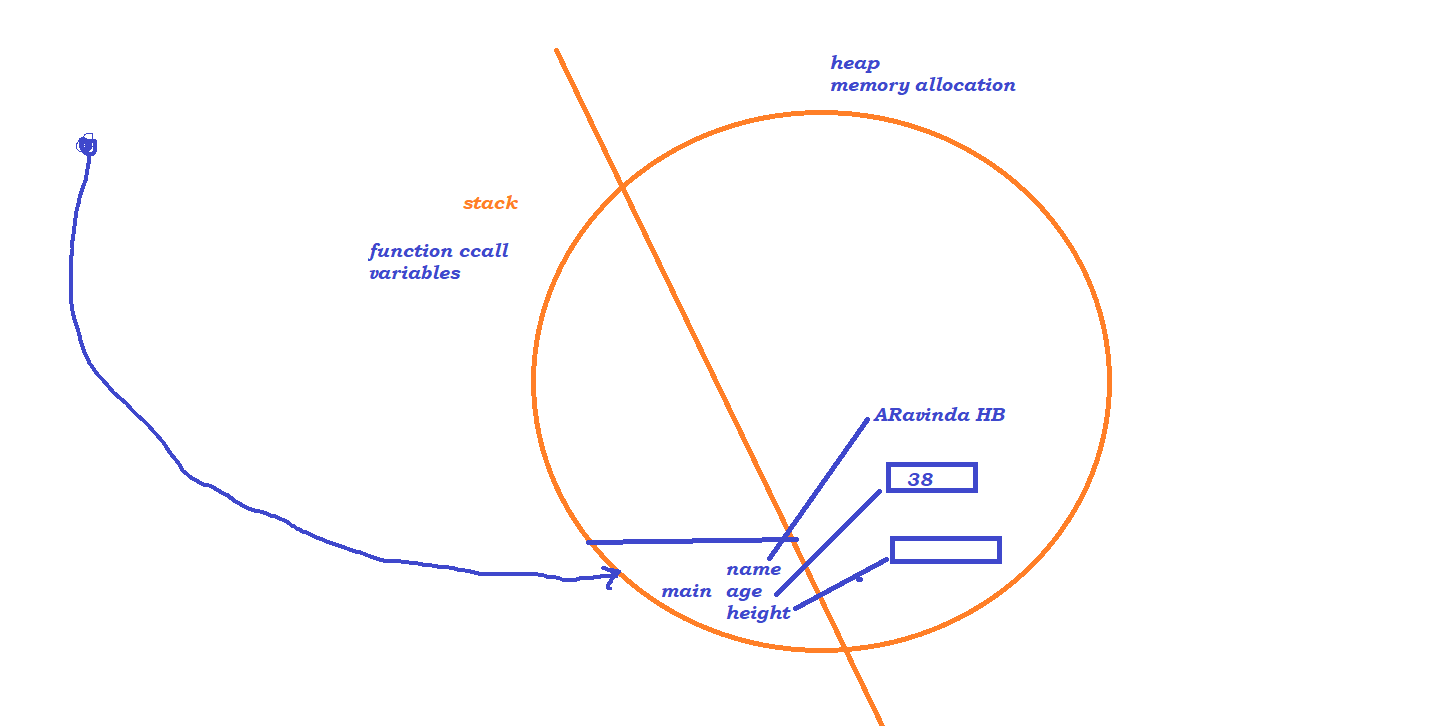
Variables :

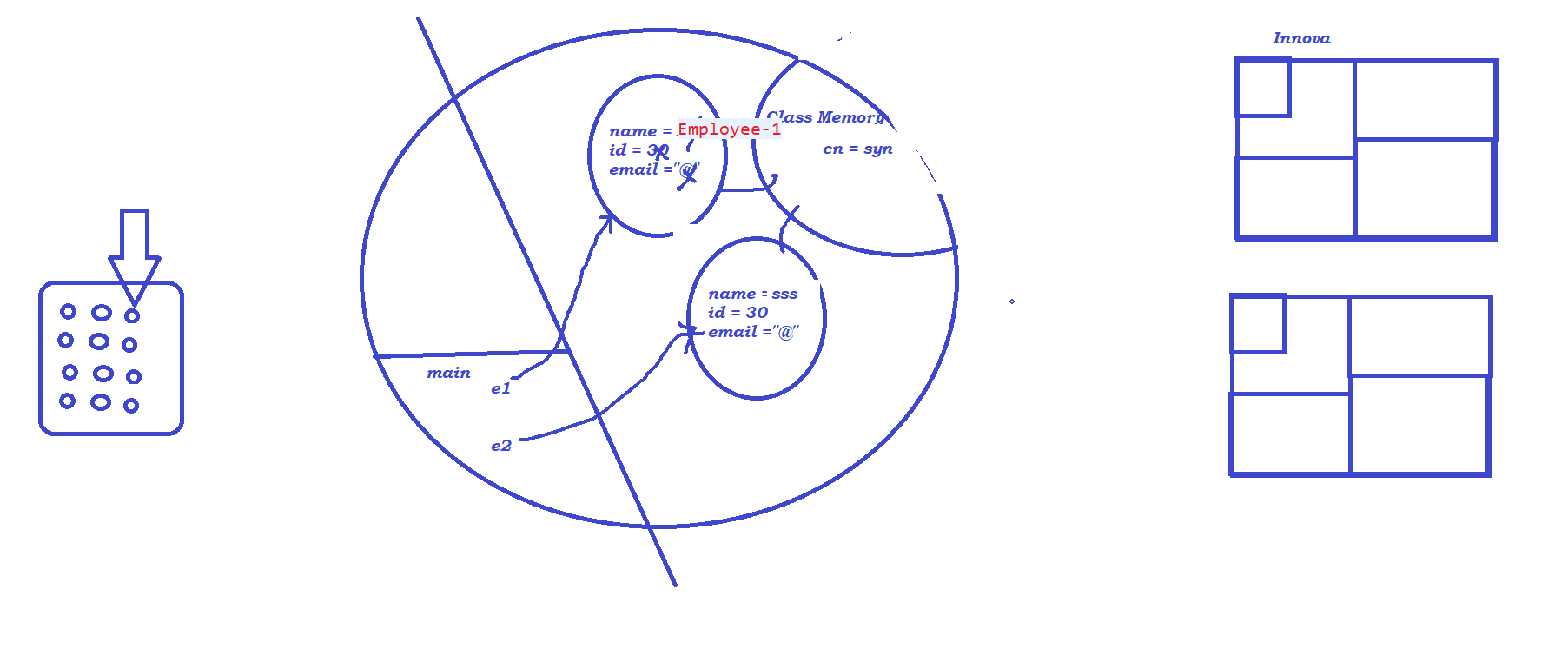


String name = “Aravinda”

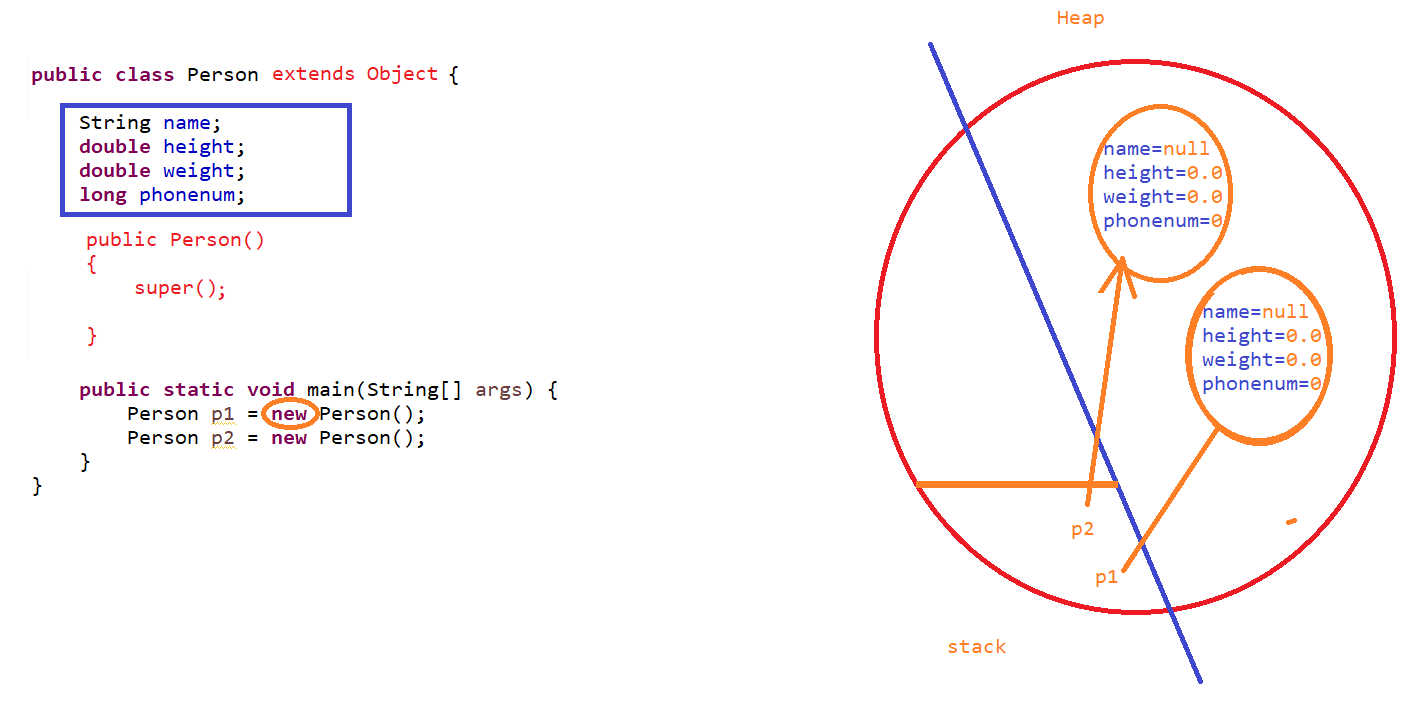
int age = 38

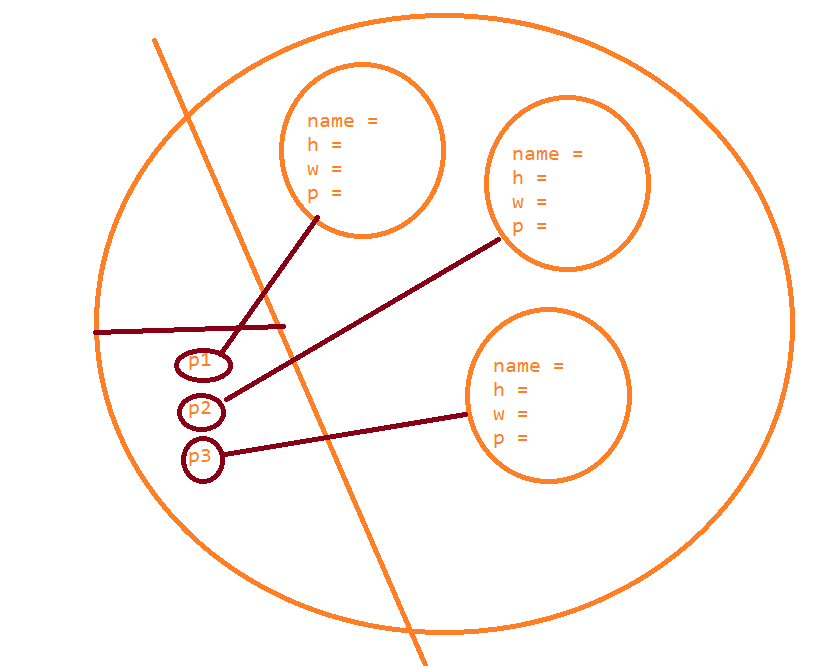
double height = 5.8



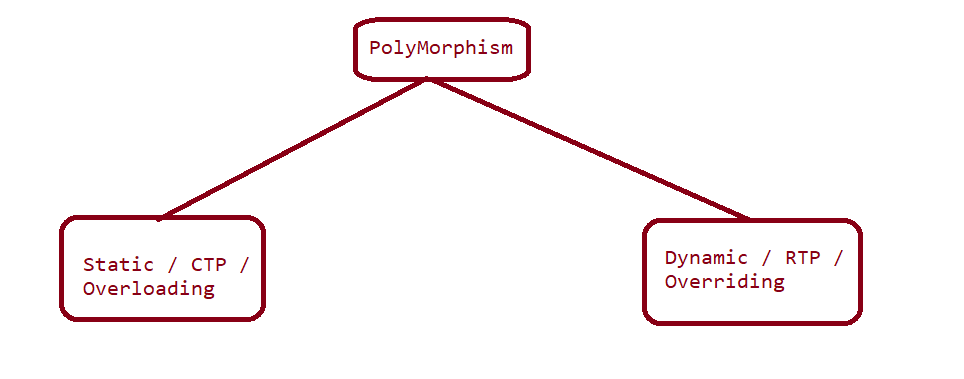


Constructors:





Polymorphism :



# Overloading

One or more function having **same name** *but different signature* in a same class,

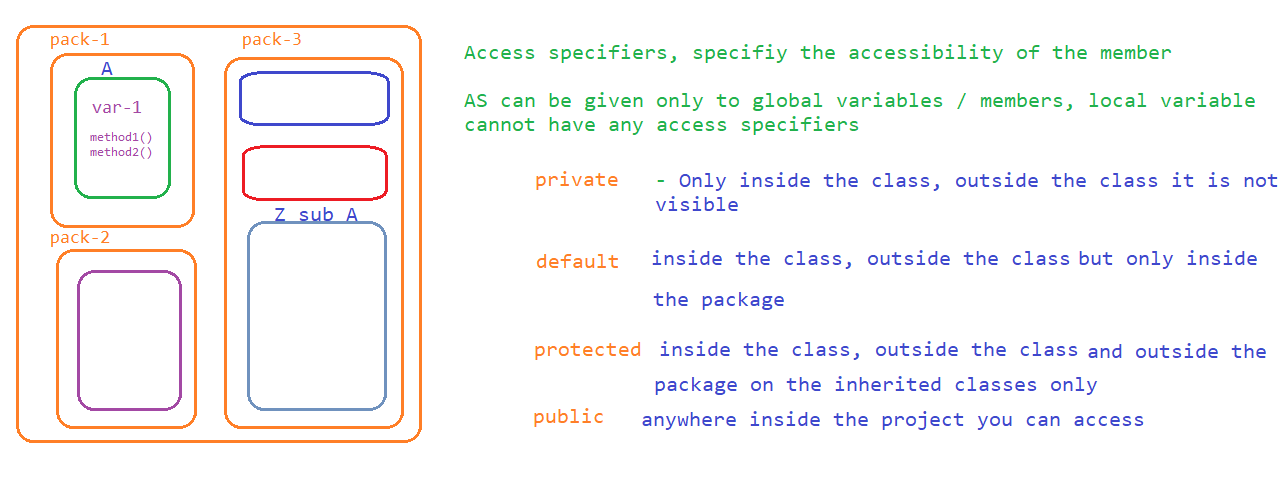
* Type of arguments
* Number of arguments
* Return type is not part of method signature

# Overriding

2 methods having same name same signature one is in parent class and another is in child class.

In case of overriding we give new implementation to the existing function.

## Access Specifiers



## Access Modifiers

* static
* final
* synchronized
* abstract

Initialization Blocks :

1. Static Initialization Block

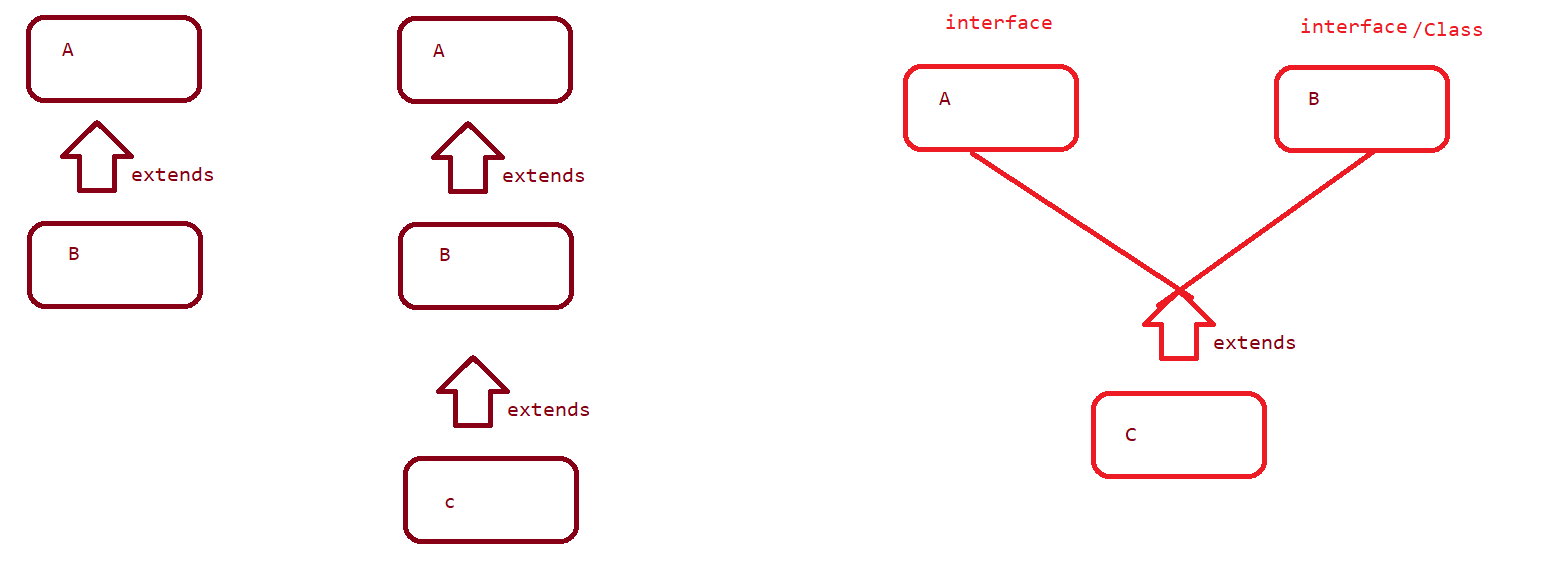
2. Instance Initialization Blocks

Abstract Classes:

Class with implementation gap

Interfaces

Inheritence

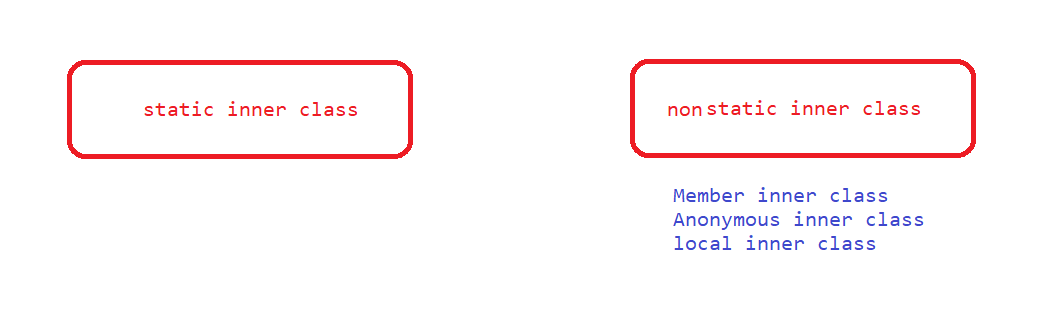


looping and conditional Statements

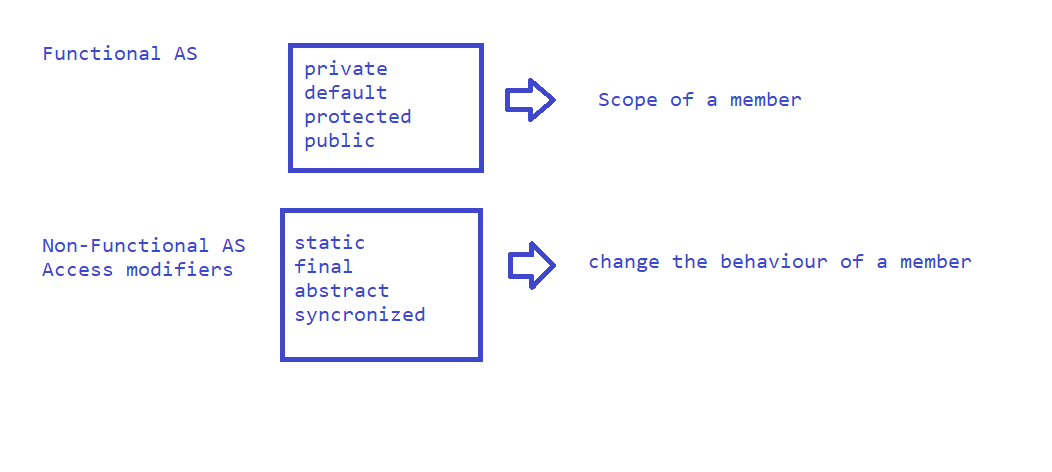
* for
* while
* do while
* foreach
* if
* if,else
* if elseif else
* switch

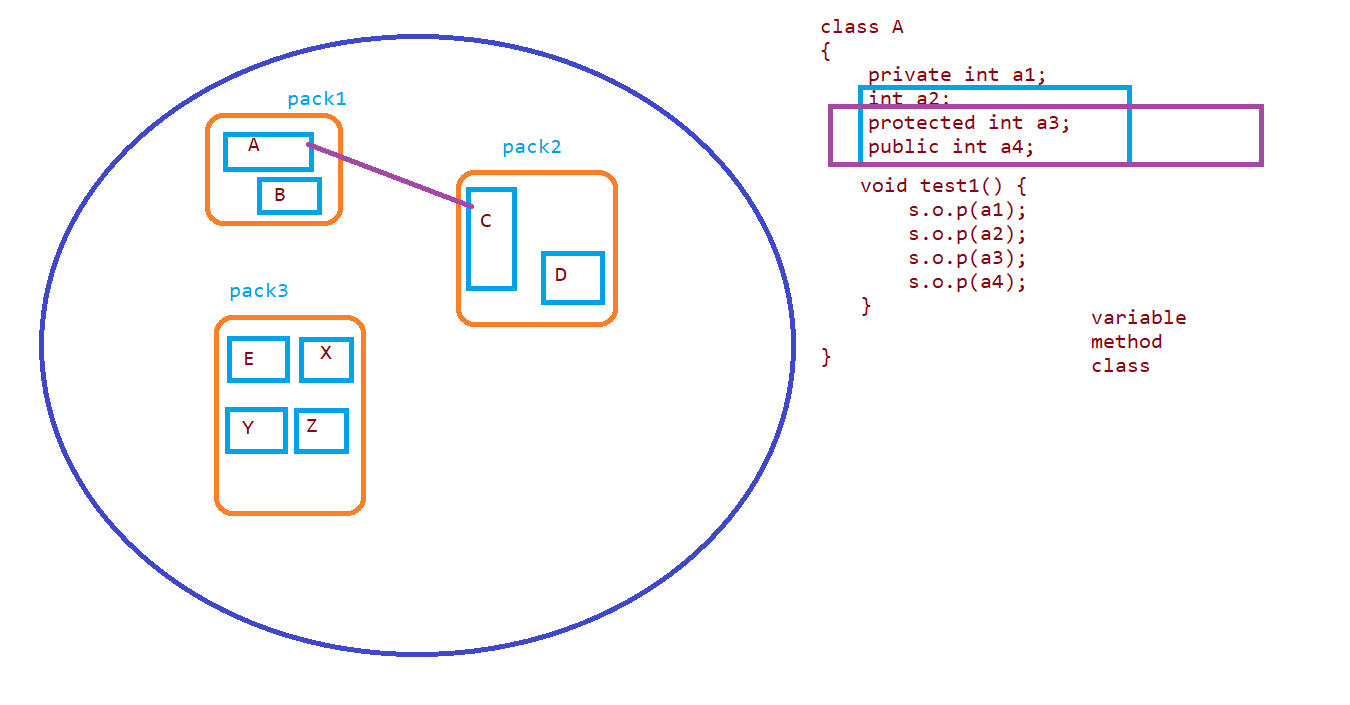
Inner Classes

Anonymous inner class

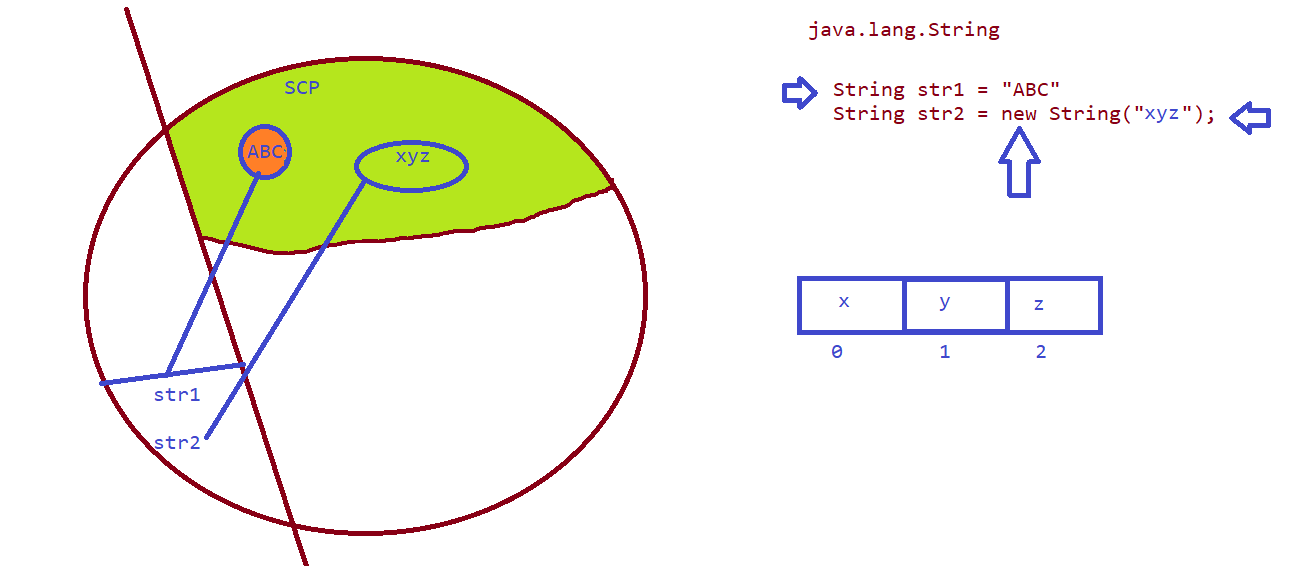


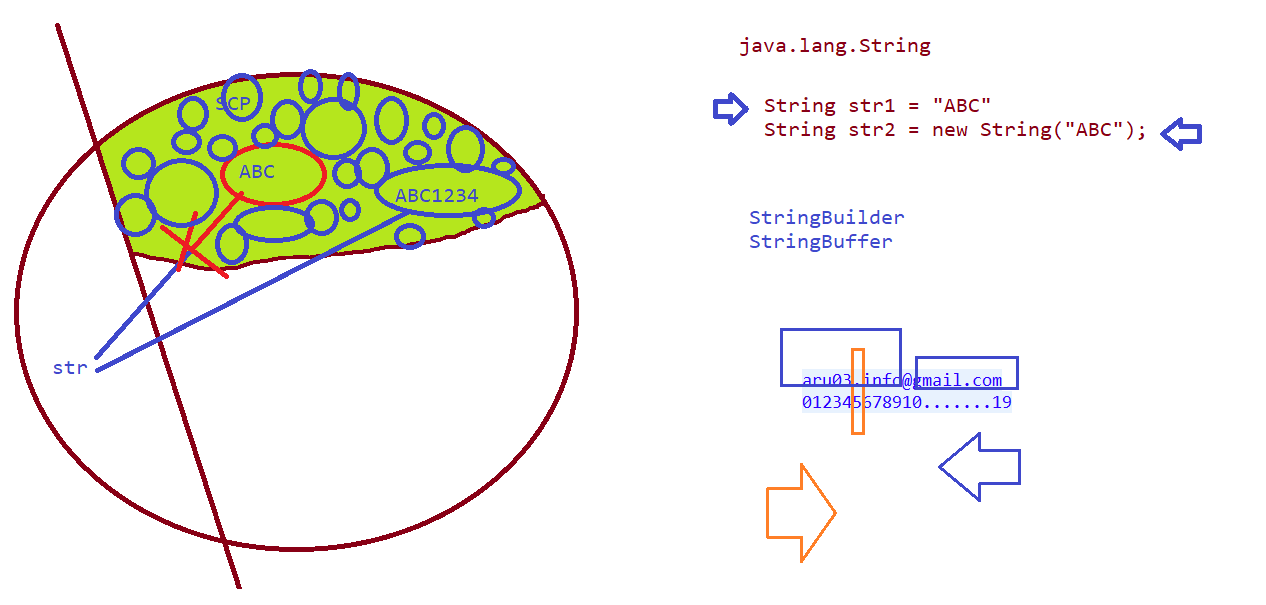
Access specifiers and modifiers :



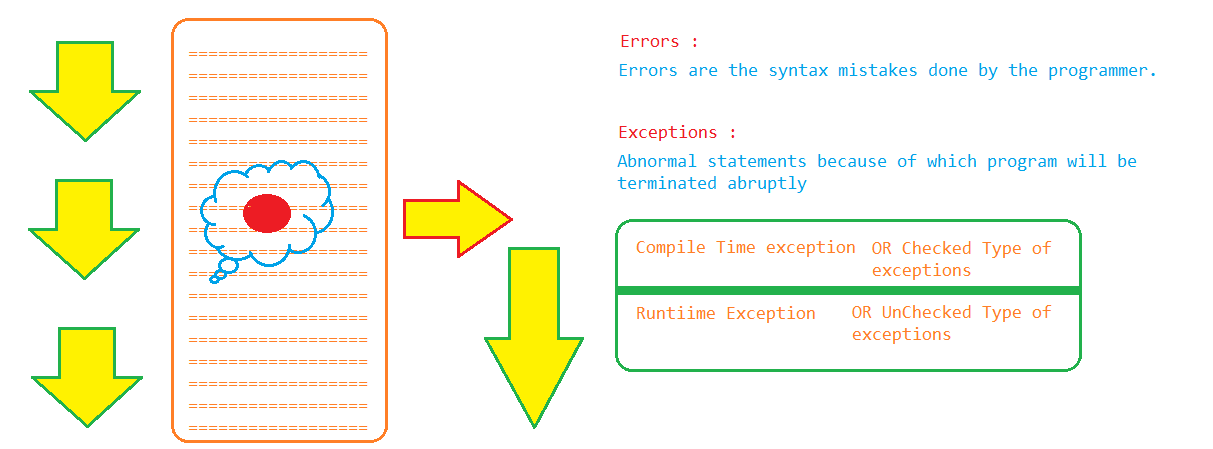


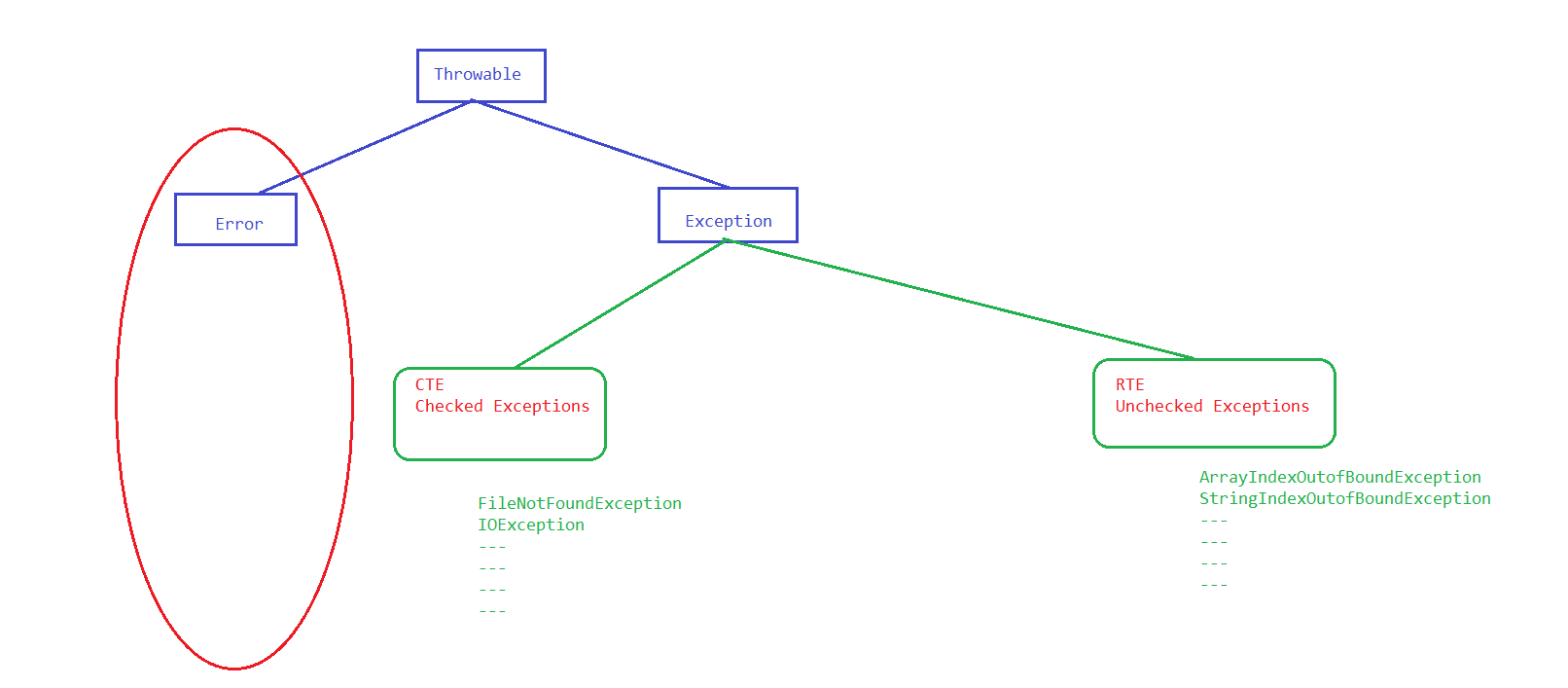
Strings :

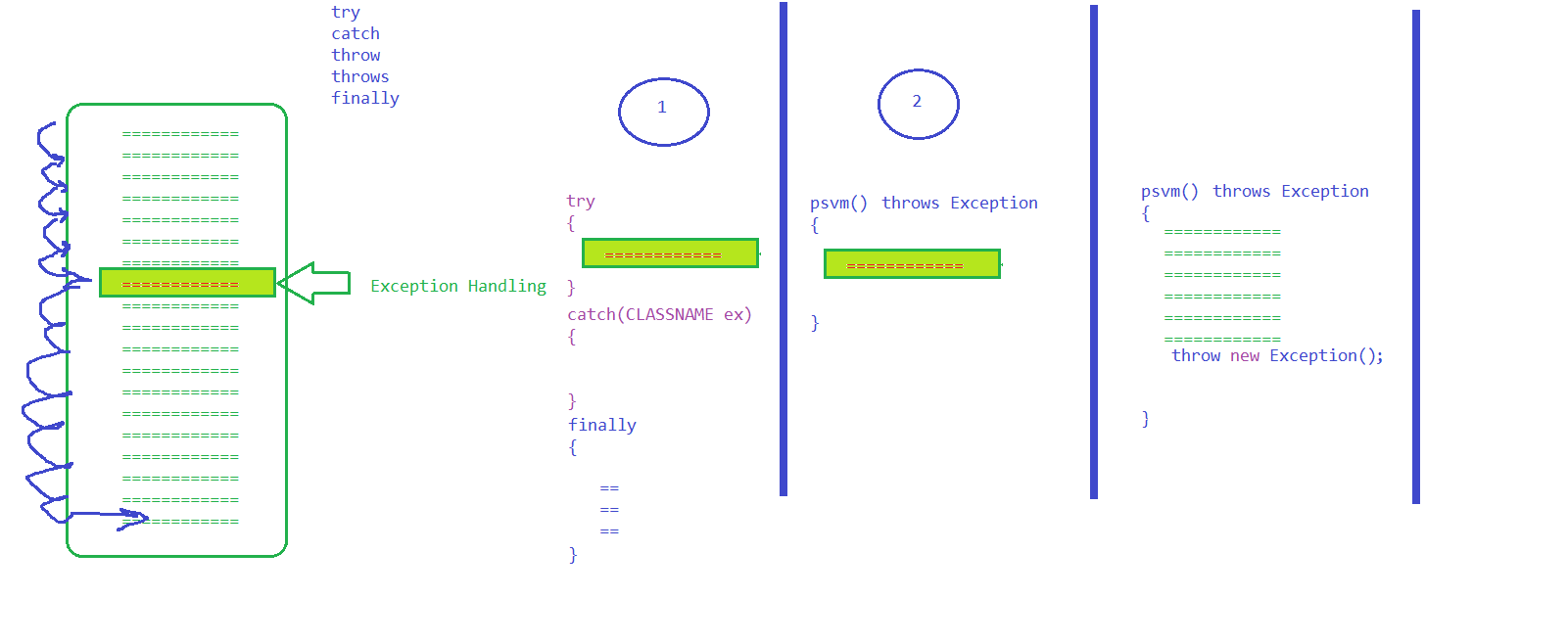




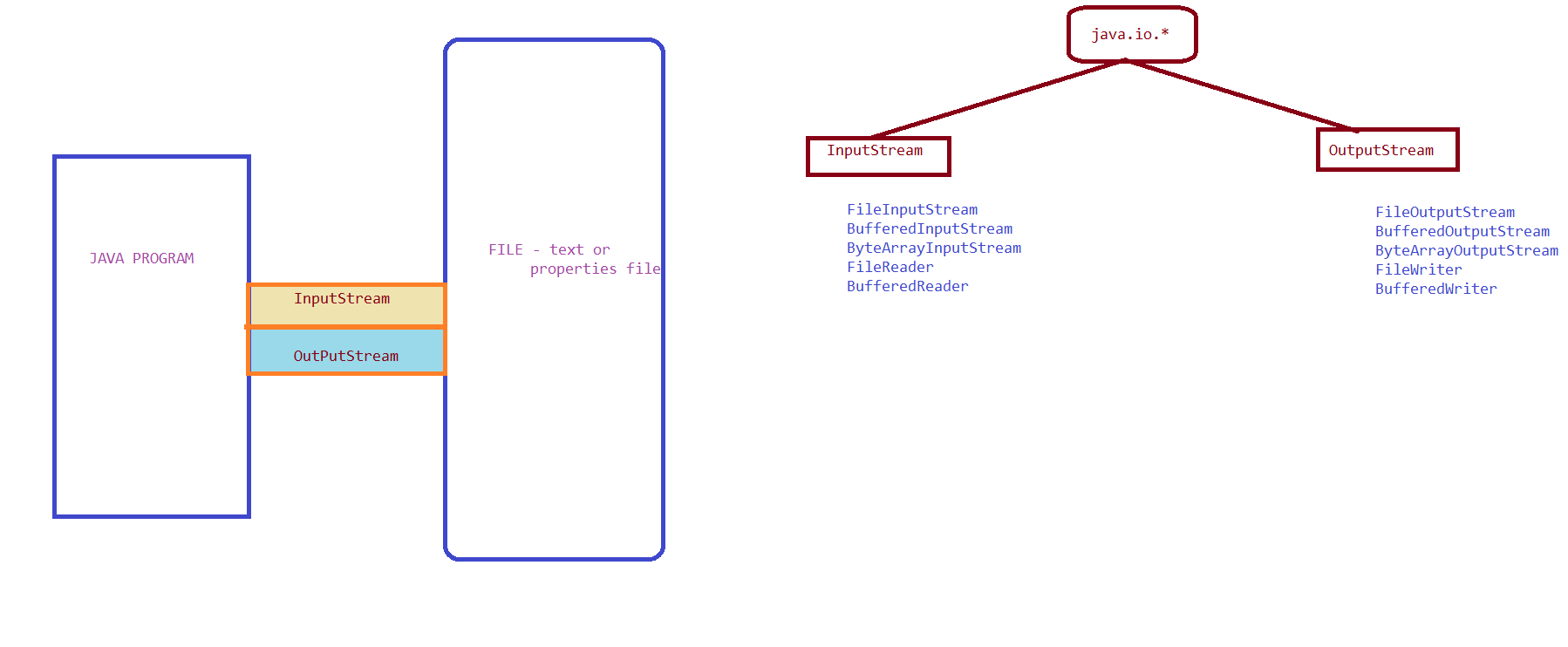
Exceptions:

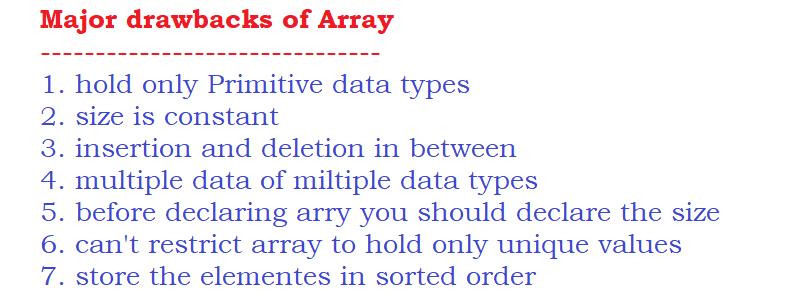




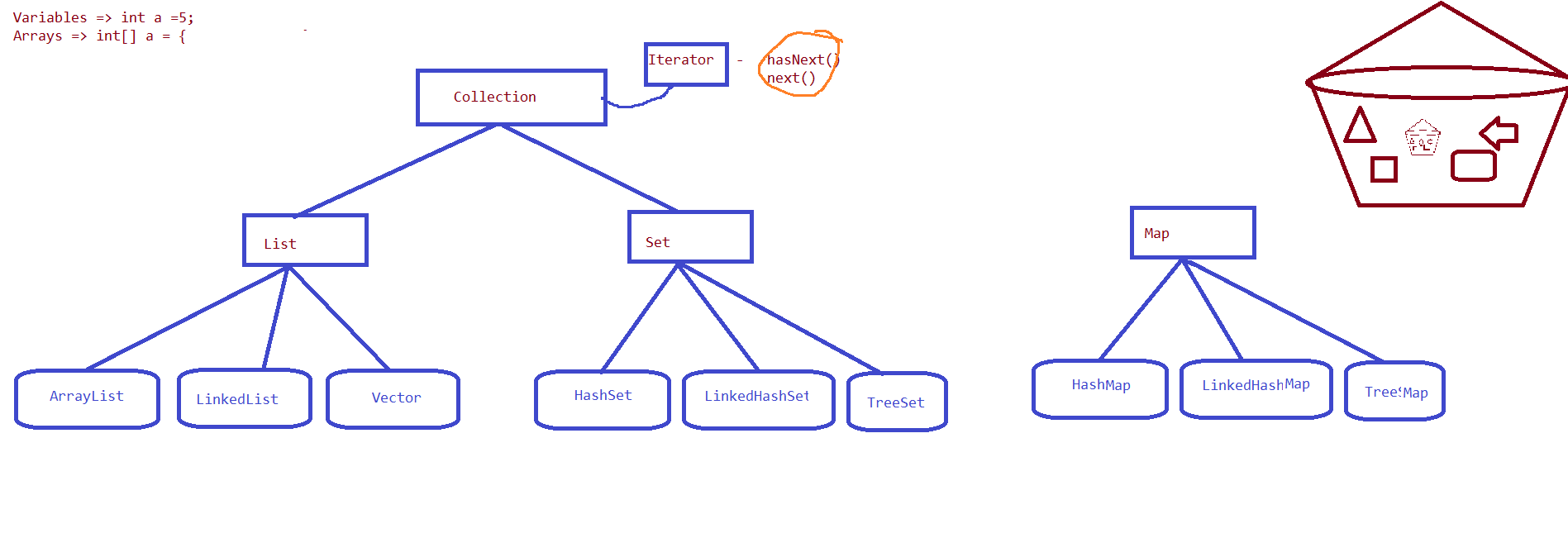


File Handling





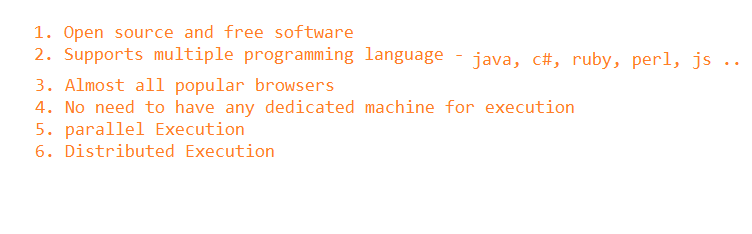
Collections:



Generics :

Parameterized Collections

Selenium:

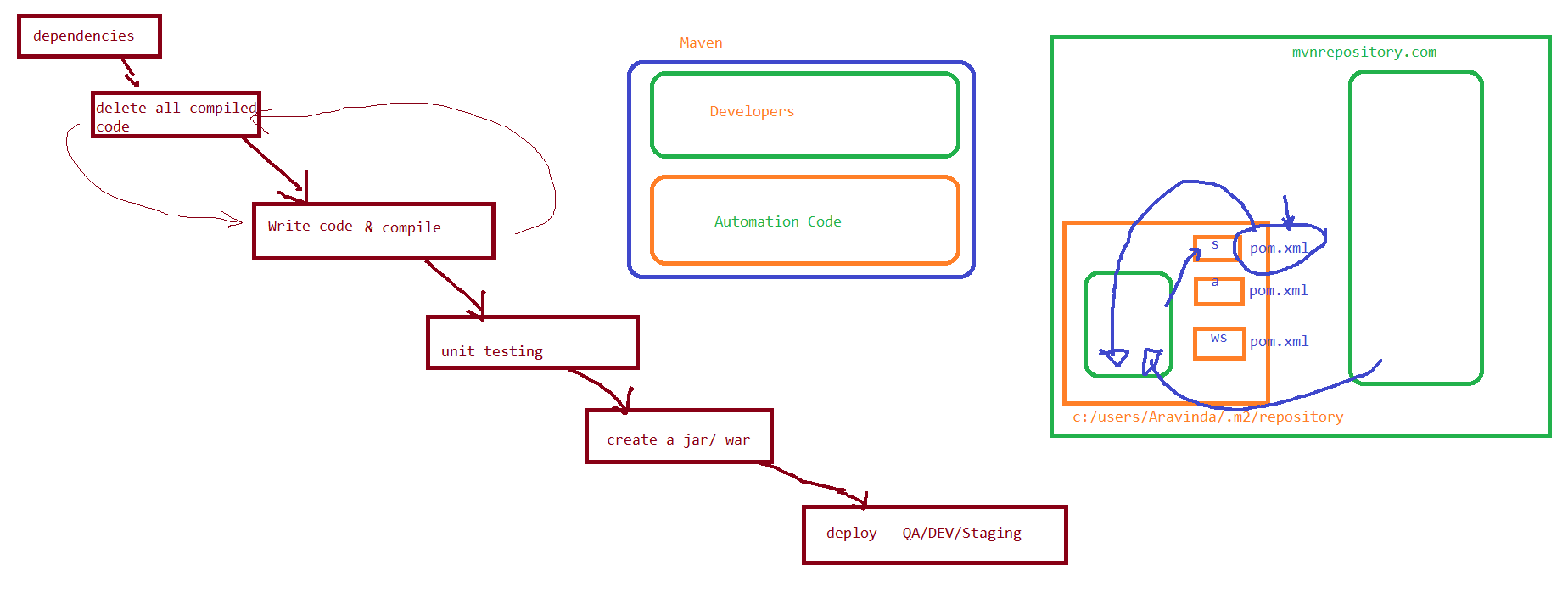


* Identification
  1. id
  2. name
  3. classname
  4. tagname
  5. linktext
  6. partiallinktext
  7. css
  8. xpath
* CSS Selector :

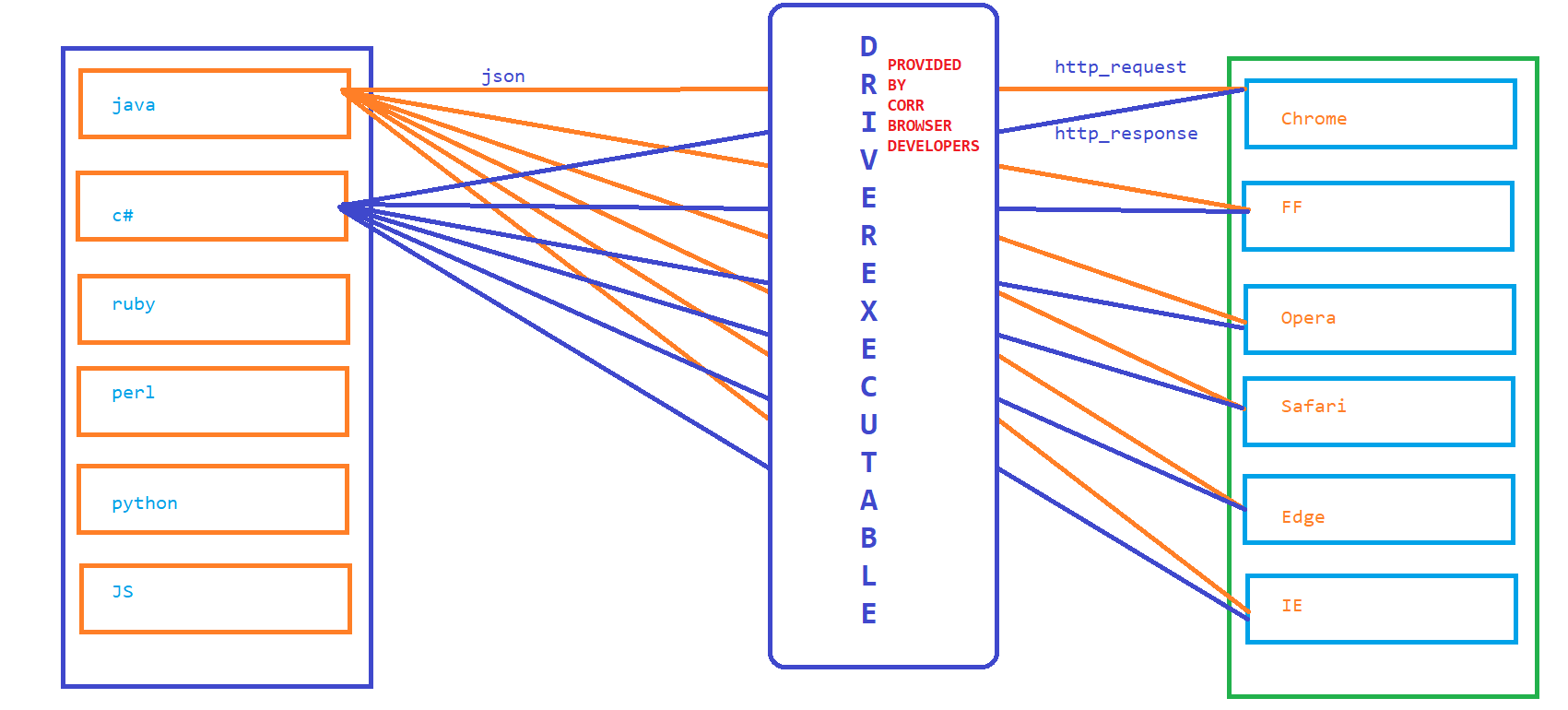
1. htmltag[attribute = ‘value’] - input[placeholder='Enter address']
2. htmltag[id = ‘value’] OR htmltag#idvalue OR #idvalue
   1. input#autocomplete OR #autocomplete
3. htmltag[class = ‘value’] OR htmltag.classValue OR .classvalue
   1. input.form-control OR .form-control
4. Parent to child
   1. expression of parent > child expression OR TAG
   2. a#loginButton > div

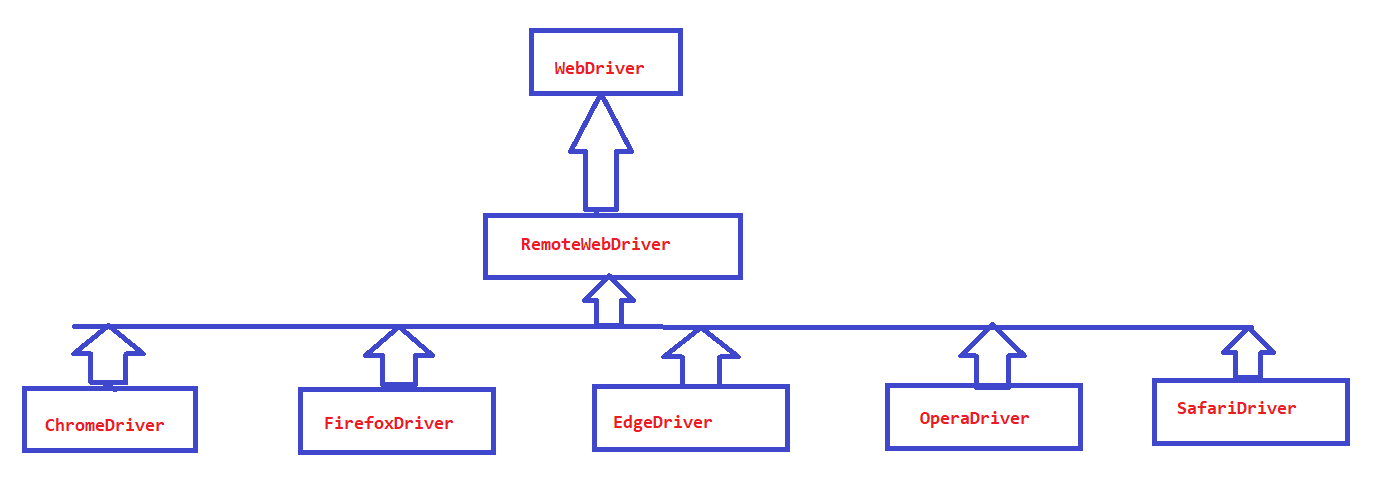
* XPATH
  1. **Basic xpath**
     + //htmltag[@attribure=’value’]
  2. **Xpath using functions**
     + text() - > Exact text of an element
     + EX: //label[text()='Street address']
     + contains(arg1, arg2)
       - arg1 – attribute or text function
       - arg2 – partial value of attribute or text function
       - Attribute : //htmltag[contains(@attribute,’partialvalue’)]
         * EX: //img[contains(@src,'timer')]
       - Function: //htmltag[contains(functionname,’partialvalue’)]
         * EX: //label[contains(text(),'Keep')]
     + starts-with(arg1, arg2)
       - arg1 – attribute or text function
       - arg2 – starting value of attribute or text function
       - Attribute :
         * //\*[starts-with(@id,'ext-gen')]
       - Function:
         * //h3[starts-with(text(),'About')]
  3. **Logical operators**
     + AND -> //htmltag[@att1=’val1’ and @att2=’val2’]
       - //td[(@class='current day' or @class='wd day' or @class='we day') and text()='3']
     + OR ->//htmltag[@att1=’val1’ or @att2=’val2’]
     + NOT -> htmltag[@att1=’val1’ and not @att2=’val2’]
       - //td[text()='6' and not(@class='past day')]
  4. **Traverse from parent to child**
     + **Immediate Child:** //a[@id='loginButton']/div
     + **Somewhere :** //td[@id='loginButtonContainer']//div[text()='Login ']
  5. **Traverse from child to parent**
     + Whenever we see dependent and independent elements in the application then we go with traversing from child to parent
     + STEP-1 : Write an xpath for independent element
       - **//th[text()='Directed by']**
     + STEP-2 : Put the complete expression inside a square bracket(which is written in STEP-1 except // )
       - **//[th[text()='Directed by']]**
     + STEP-3 : Specify the parent HTML TAG
       - **//tr[th [text()='Directed by']]**
     + STEP-4 : Traverse to the dependent Element
       - **FINAL : //tr[th[text()='Directed by']]//a**
       - **EX-2 :** //tr[td[div[div[@title='Task203']]]]/td[@class='selection']
  6. **Axes Functions**
     + **Traversing between following-sibling**
       - //th[text()='Directed by']/**following-sibling::**td
     + **Traversing between preceding-sibling**
       - //tr[th[text()='Starring']]/**preceding-sibling::**tr
       - //tr[th[text()='Starring']]/preceding-sibling::tr[th[text()='Directed by']]
     + **Traversing till the beginning of the Page** 
       - //a[text()='Best Action']/following::a
     + **Traversing till the end of the Page** 
       - //a[text()='Best Action']/preceding::a
     + **Traversing to parent**
       - //th[text()='Produced by']/parent::tr
     + **Traversing to child**
       - //th[text()='Produced by']/parent::tr/child::td
     + **Traversing to ancestor**
       - //div[text()='Task203']/ancestor::tr/td[@class='selection']
       - //h3[text()='Amazing Goa Flight Inclusive Deal 3N']/ancestor::div[contains(@class,'packageListing')]//p[contains(@class,'font26')]

MAVEN :

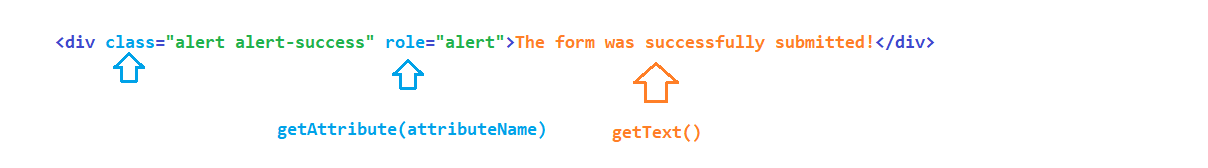


Architecture

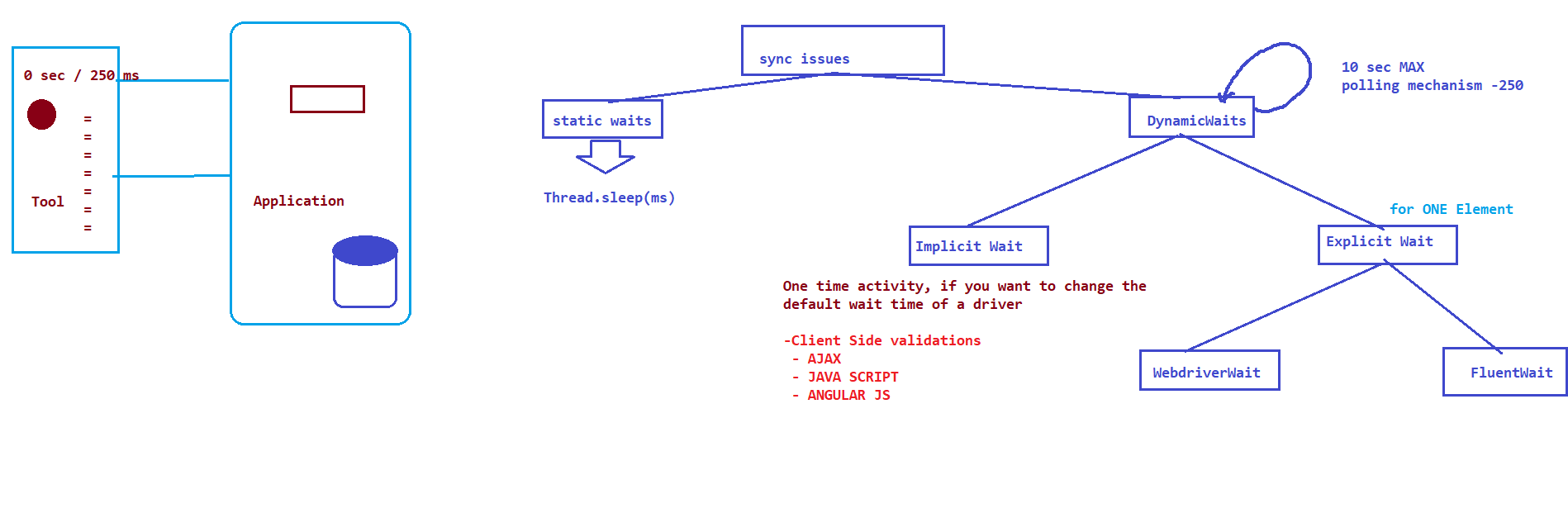




Getting Text At Runtime OR Getting Attribute at Runtime :



Sync Issues



1. WebDriver Wait

STEP1 : Create an object to WebdriverWait by passing max wait time

STEP2 : call until method present inside wait object

STEP3 : Use ExpectedConditions class provided selenium lib and call appropriate methods

2. Fluent Wait

STEP1 : Create a object to Fluent wait by passing input

INPUT 🡪1. what is the webelement

2. what is the max timeout

3. what is the polling time

4. what are the exceptions to be ignored!!!

STEP2 : Create an object to Function Interface, and implement

apply method ( keep your own wait logic )

STEP3 : call until method present inside wait object and pass function object

Browser Operations :

back –

forward

current url

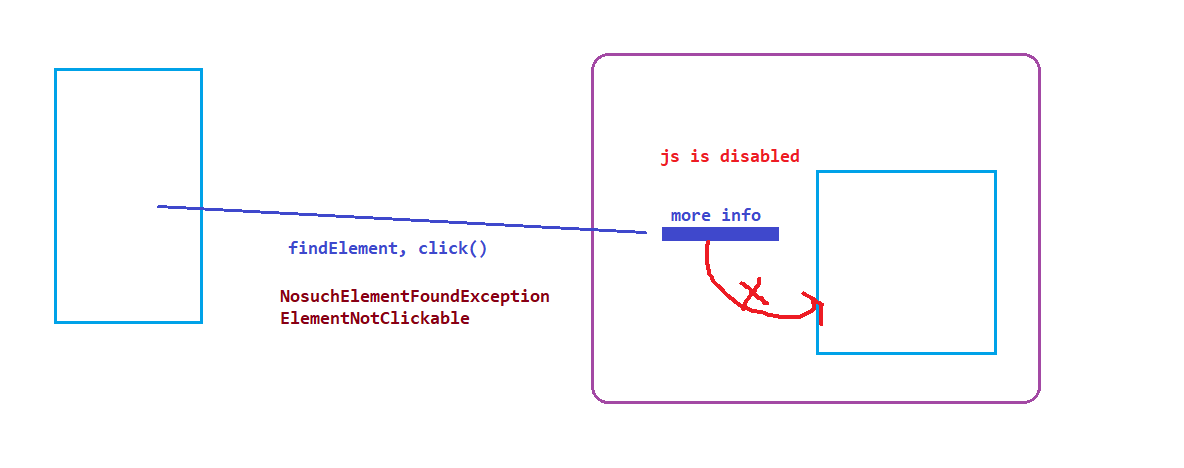
windowed

titile

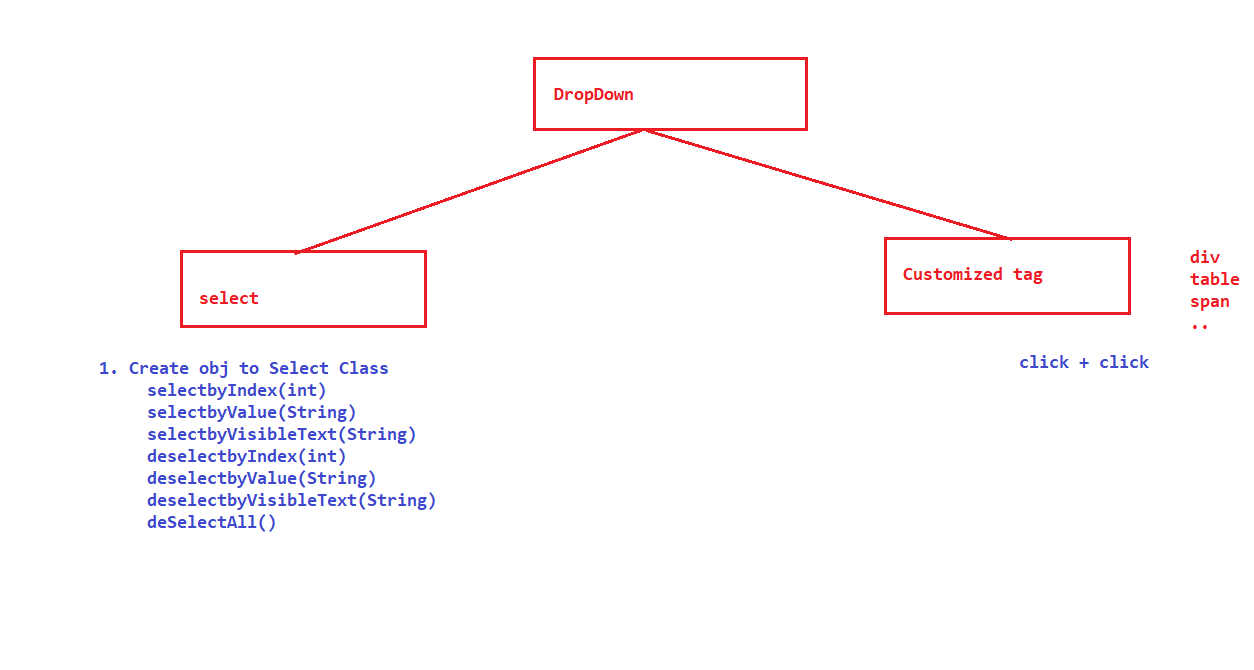
refresh

maximize

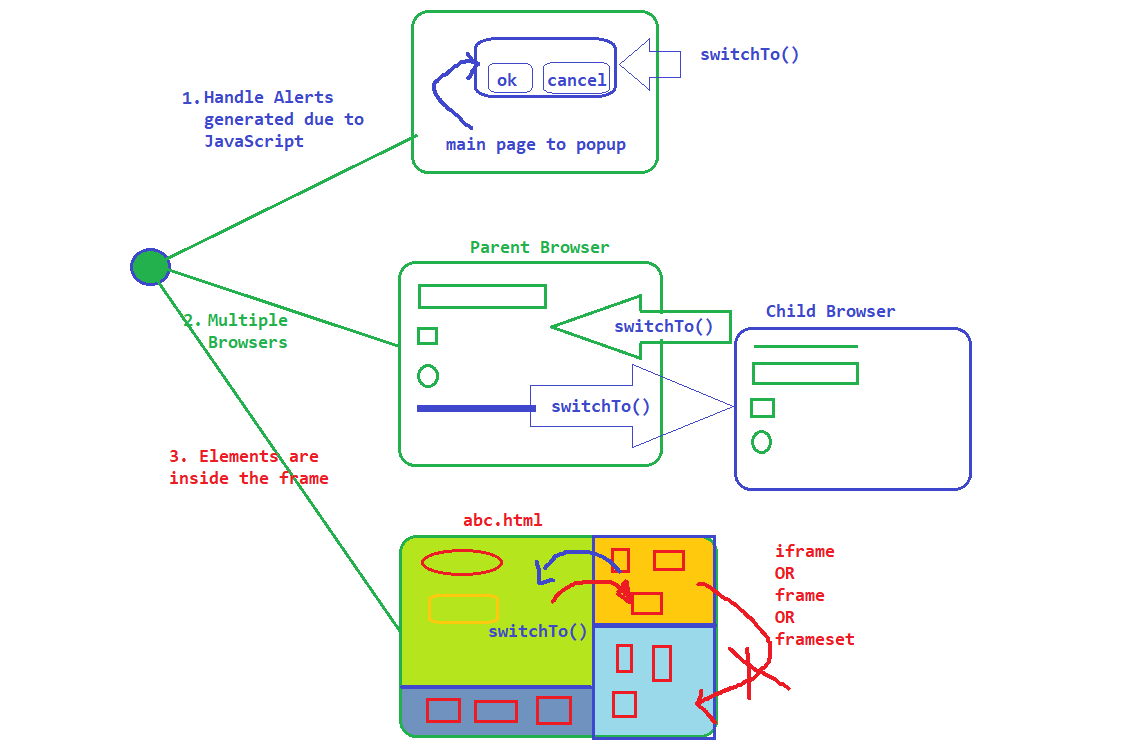
Actions : Used to perform Exact KB and Mouse Operations



DropDown Handling in selenium



SWITCH-TO :



driver.switchTo().frame(index);

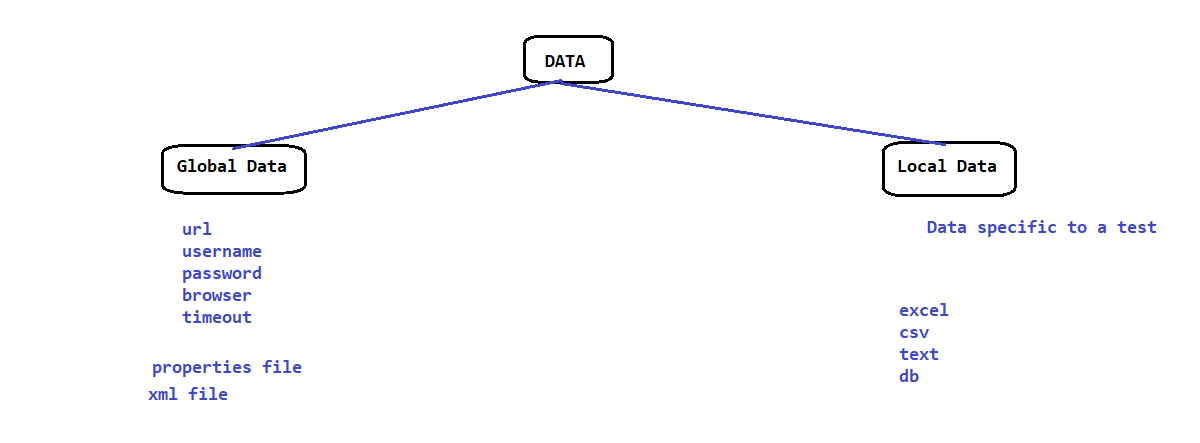
driver.switchTo().frame(id/name);

driver.switchTo().frame(webelement);

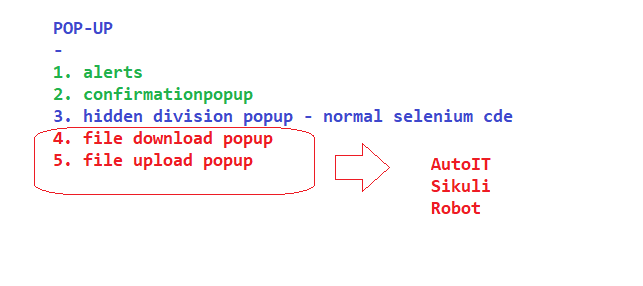
driver.switchTo().defaultContent();

\

Data driven testing :

****

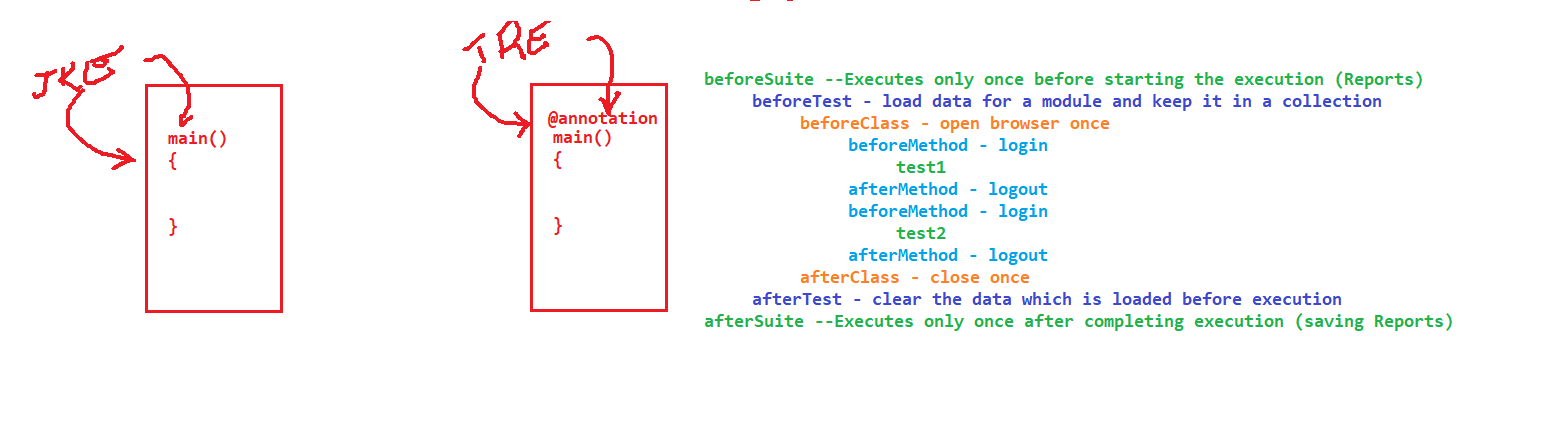
POPUP :



FRAMEWORK :

1. Function driven Automation Framework
2. Keyword driven Automation Framework
3. Data driven Automation Framework
4. Hybrid driven Automation Framework
5. TestNG –
6. POM

TESTNG:



POM

