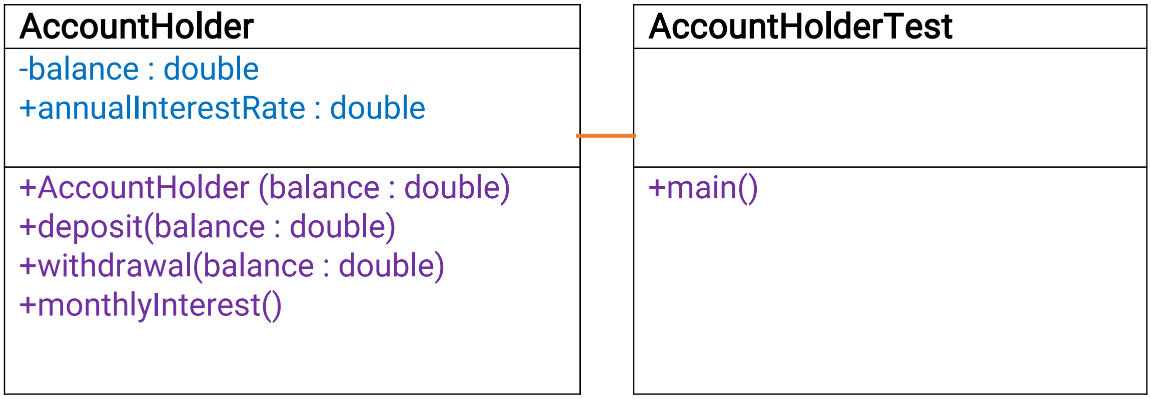
Lab 1 Objectives / General Tips - **Chapter references: Chap 3,6,8**

[UML of classes]



AccountHolder

-balance : double
+annualInterestRate : double

+AccountHolder (balance : double)
+deposit(balance : double)
+withdrawal(balance : double)
+monthlyInterest()

AccountHolderTest

+main()

[Accessing data]

-2 ways to access data

1) For static members (fields) *ClassName.staticmember* Ex. AccountHolder.annualInterestRate = .05;	2) For instance members (fields) *objectName.method()*

[The Constructor]

When object is created the *Constructor* is called automatically.

Ex.

AccountHolder accObj1 = new **AccountHolder**(balance);

[Error trappings!]

```
 if (balance < 0.0)
      throw new IllegalArgumentException("balance must be non-negative");
```

[Error trappings!]

other:

```
while (true) {
          balance = sc.nextDouble();
          if (balance < 0)
             System.out.println("Pls. reenter a positive beginning balance");
          else
            break;
}
```

[Avoiding Self-Referential assigning!]

*Use this (which refers to a current object's instance) keyword in a method to avoid self referential integrity!

```
 this.balance = balance;
```

ex.

```
// constructor, creates a new account with the specified balance
public AccountHolder(double balance) {

  this.balance = balance;    //assign local variable to class member

}
```

[Print formatting]

Use various print formatting with the format specifier % symbol followed by a converter.

Popular converters to use include:

%f -> float
%d -> int
%s -> string
%n -> newline

Ex.

```
System.out.printf("$%.2f", balance);  //print currency style
System.out.format ("%-10s%n", "Monthly balance");
```